Integration Service Platform

Version 1.8.0

Table of Contents

Chapter 1

Introduction to the Integration Service

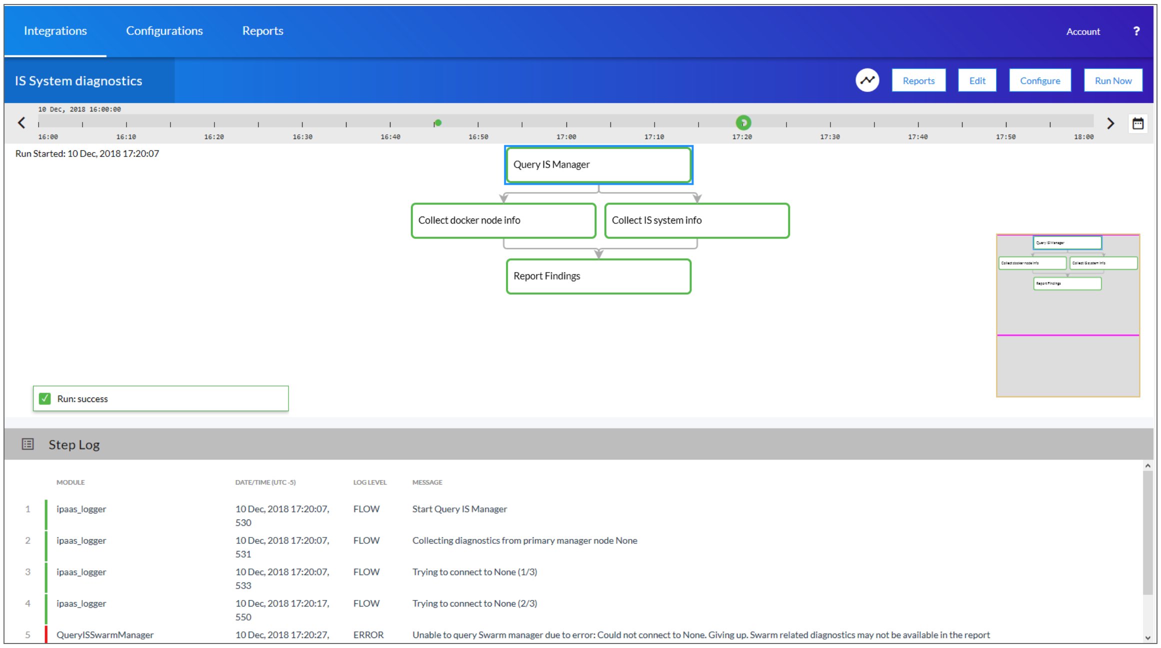
Overview

The Integration Service provides a generic platform for integrations between SL1 and third-party platforms.

This chapter covers the following topics:

What is the Integration Service?

The Integration Service enables intelligent, bi-directional communication between the ScienceLogic data platform and external data platforms to promote a unified management ecosystem. The Integration Service allows users to translate and share data between SL1 and other platforms without the need for programming knowledge. The Integration Service is designed to provide high availability and scalability.



Run Started: 10 Dec, 2018 17:20:07

☑ Run: success

```
   MODULE               DATE/TIME (UTC -5)       LOG LEVEL   MESSAGE
1  ipaas_logger         10 Dec, 2018 17:20:07,   FLOW        Start Query IS Manager
                        530
2  ipaas_logger         10 Dec, 2018 17:20:07,   FLOW        Collecting diagnostics from primary manager node None
                        531
3  ipaas_logger         10 Dec, 2018 17:20:07,   FLOW        Trying to connect to None (1/3)
                        533
4  ipaas_logger         10 Dec, 2018 17:20:17,   FLOW        Trying to connect to None (2/3)
                        550
5  QueryISSwarmManager  10 Dec, 2018 17:20:27,   ERROR       Unable to query Swarm manager due to error: Could not connect to None. Giving up. Swarm related diagnostics may not be available in the report
```

The key elements of the Integration Service user interface include the following:

- ***Steps***. A step is a generic Python class that performs an action. Steps accept arguments and can be re-used. The arguments tell the step which variables and values to use when executing. Steps can also pass results to a subsequent step.
- ***Integration Applications***. An integration application is a JSON object that includes all the information required for executing an integration on the Integration Service platform. An integration application includes a list of steps and metadata for those steps. Each step will execute a single action and pass the results to subsequent, dependent steps. The parameters for each step are also defined in the application and can be provided either directly in the step or in the parent integration application.
- ***Configurations***. A configuration is a stand-alone JSON file that contains a set of configuration variables. Configurations live on the Integration Service system and can be accessed by *all integration applications and their steps.*

What is a Step?

In an Integration Service system, a ***step*** is a generic Python class that performs a single action, such as caching device data:

CacheAllDeviceData

Steps accept arguments called *parameters*. These arguments specify the values, variables, and configurations to use when executing the step. Parameters allow steps to accept arguments and allow steps to be re-used in multiple integrations. For example, you can use the same QueryREST step to query both the local system and another remote system; only the arguments, (hostname, username, and password) change.

A step can pass the data it generates during execution to a subsequent step. A step can use the data generated by another step.

The Integration Service system analyzes the required parameters for each step and alerts the user if any required parameters are missing before the Integration Service can run the step.

Steps are grouped into the following types:

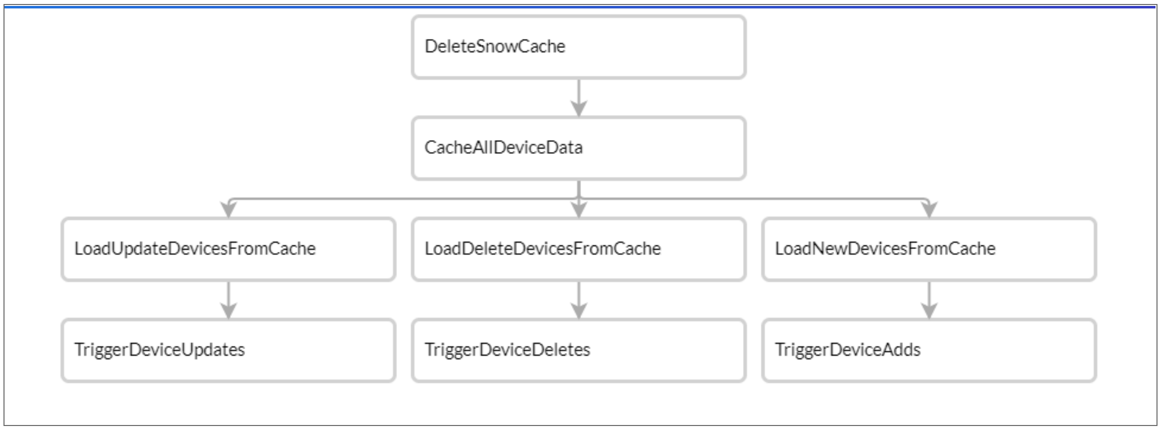
- ***Standard***. Standard steps do not require any previously collected data to perform. Standard steps are generally used to generate data to perform a transformation or a database insert. These steps can be run independently and concurrently.
- ***Aggregated***. Aggregated steps require data that was generated by a previously run step. Aggregated steps are not executed by the Integration Service until all data required for the aggregation is available. These steps can be run independently and concurrently.
- ***Trigger***. Trigger steps are used to trigger other integration applications. These steps can be configured to be blocking or not.

A variety of generic steps are available from ScienceLogic, and you can access a list of step parameters in the ***API endpoint***.

What is an Integration Application?

In the Integration Service, an ***integration application*** is a JSON file that specifies which steps to execute and the order in which to execute those steps. An integration application also defines variables and provides arguments for each step.

The following is an example of an integration application:



Integration application JSON objects are defined by configuration settings, steps that make up the integration, and application-wide variables to use as parameters for each step. The parameters of each step can be configured dynamically, and each step can be named uniquely while still sharing the same underlying class, allowing for maximum re-use of code.

Integration applications can be executed through the REST API and are processed as an asynchronous task in the Integration Service. During processing the user is provided a unique task ID for the application and each of its tasks. Using the task IDs, the user can poll for the status of the integration application and the status of each individual running step in the integration application.

Executing an integration application from the REST API allows the user to dynamically set one-time parameter values for the variables defined in the integration.

The required parameters of integration applications are strictly enforced, and the Integration Service will refuse to execute the integration application if all required variables are not provided.

What is a Configuration?

Configuration variables are defined in a stand-alone JSON file called a ***configuration*** that lives on the Integration Service system and can be accessed by all integration applications and their steps.

Each global variable is defined as a JSON object in the configuration. Typically, a configuration object looks like the following:

```
{"name": "var_name", "value":"var_value", "encrypted": true}
```

Each global variable in the configuration has the option of being encrypted. The values of encrypted variables are encrypted within the Integration Service upon upload through the REST API.

Creating and Saving Integration Service Components

Instead of using the Integration Service user interface, you can create steps, integration applications, and configurations in your own editor and then upload them using the API or the command line interface (CLI).

For more information, see the ***Integration Service for Developers*** manual.

Chapter

2

Installing and Configuring the Integration Service

Overview

This chapter describes how to install and configure the Integration Service, and also how to set up security for the service.

This chapter covers the following topics:

Prerequisites for the Integration Service

To work with the Integration Service, ScienceLogic recommends that you have knowledge of the following:

- vi or another text editor
- Linux
- Docker. More information on using the Docker command line can be found in the ***Helpful Docker Commands*** section and at https://docs.docker.com/engine/reference/commandline/cli/.
- Python
- Couchbase

In addition, you must give your Docker Hub ID to your ScienceLogic Customer Success Manager to enable permissions to pull the containers from Docker Hub. The Integration Service requires the **docker-ce 18.06** version of Docker.

System Requirements

The Integration Service has the following system requirements:

- 8 CPUs
- 24 GB total RAM
- 100 GB total storage

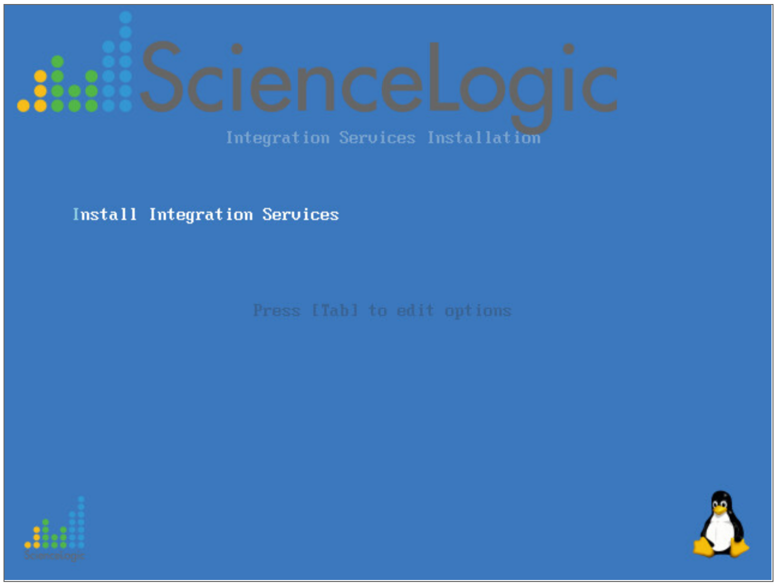
You should also use an **ap2** version of 5.54.9 or later of the new user interface for SL1. For more information, see the ***Introduction to the New User Interface*** manual.

Installing the Integration Service

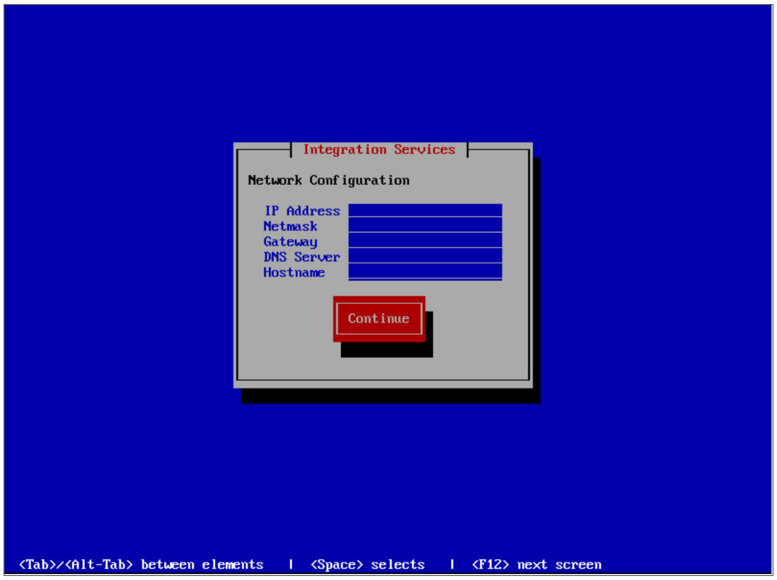
To install the Integration Service:

1. Download the latest Integration Service ISO file to your computer.

2. Using your hypervisor or bare-metal (single-tenant) server of choice, mount and boot from the Integration Service ISO. The Integration Service Installation window appears:


ScienceLogic
Integration Services Installation
Install Integration Services
Press [Tab] to edit options

3. Select *Install Integration Service*. After the installer loads, the **Network Configuration** window appears:

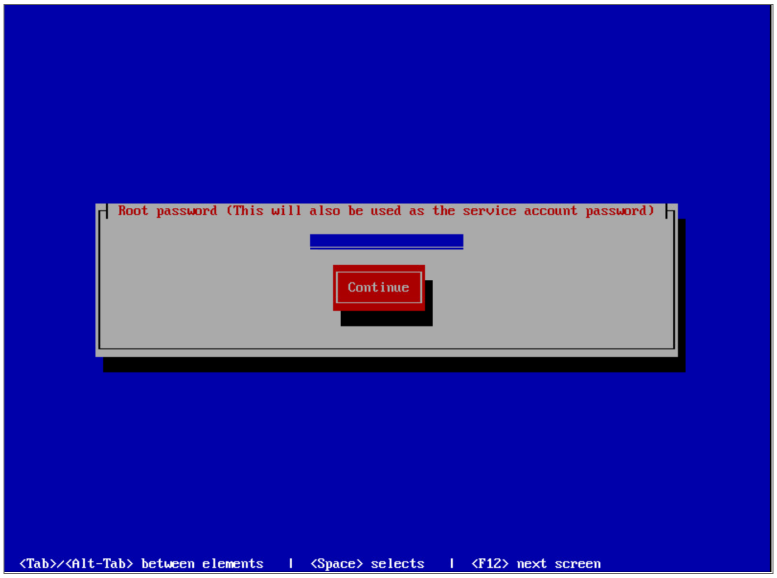


```
<Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

4. Complete the following fields:

    - ***IP Address***. Type the primary IP address of the Integration Service server.
    - ***Netmask***. Type the netmask for the primary IP address of the Integration Service server.
    - ***Gateway***. Type the IP address for the network gateway.
    - ***DNS Server***. Type the IP address for the primary nameserver.
    - ***Hostname***. Type the hostname for the Integration Service.

5. Click **[Continue]**. The **Root Password** window appears:



```
Root password (This will also be used as the service account password)
```

```
<Tab>/<Alt-Tab> between elements   |  <Space> selects   |  <F12> next screen
```

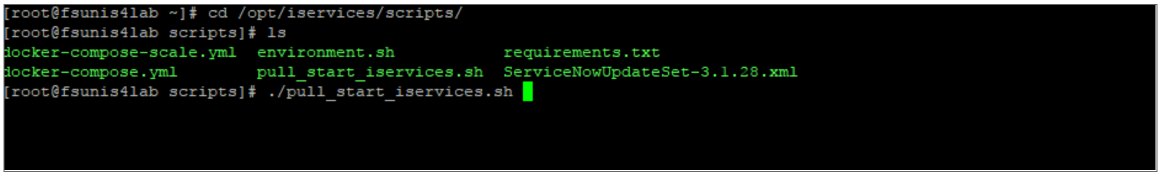
6. Type the password you want to set for the root user on the Integration Service host and press the "Enter" key.

NOTE: The password cannot contain spaces.

7. Type the password for the root user again and press the "Enter" key. The Integration Service installer runs, and the system reboots automatically.
8. Click **[Save]**.
9. SSH into the newly-installed system using PuTTY or a similar application.
10. To start services, go to opt/iservices/scripts.

11. Execute the following command:

    ```
    ./pull_start_iservices.sh
    ```

    ```
    [root@fsunis4lab ~]# cd /opt/iservices/scripts/
    [root@fsunis4lab scripts]# ls
    docker-compose-scale.yml  environment.sh           requirements.txt
    docker-compose.yml        pull_start_iservices.sh  ServiceNowUpdateSet-3.1.28.xml
    [root@fsunis4lab scripts]# ./pull_start_iservices.sh
    ```

12. Navigate to the Integration Service user interface using your browser. The address of the Integration Service user interface is:

    https://[*IP address entered during installation*]

13. Log in with the default username of *isadmin* and the password you specified above.

To verify that your stack is deployed, view your Couchbase logs by executing the following command using PuTTY or a similar application:

```
docker service logs --follow iservices_couchbase
```

If no services are found to be running, run the following command to start them:

```
docker stack deploy -c docker-compose.yml iservices
```

To add or remove additional workers, run the following command:

```
docker service scale iservices_steprunner=10
```

Upgrading the Integration Service

If you are already running the Integration Service, perform the following steps to update the service from RPM:

1. Download the RPM and copy the RPM file to the Integration Service system.
2. Either go to the console of the Integration Service system or use SSH to access the server.
3. Log in as ***isadmin*** with the appropriate (root) password. You must be root to upgrade the RPM file.
4. Type the following at the command line:

    ```
    rpm -Uvh full_path_of_rpm
    ```

    where:

    - *full_path_of_rpm* is the full path and name of the RPM file, such as **sl1-integration-services-1.x.0-1.x86_64**.

5. If the upgrade process recommends restarting Docker, run the following command:

   ```
   systemctl restart docker
   ```

6. After the RPM is installed, re-deploy the Docker stack to update the containers:

   ```
   docker stack deploy -c /opt/iservices/scripts/docker-compose.yml iservices
   ```

7. After you re-deploy the Docker stack, the services automatically update themselves. Wait a few minutes to ensure that all services are updated and running before using the system. You can use the visualizer at port 8080 to monitor the progress of the updates .
8. To view updates to each service, type the following at the command line:

   ```
   docker ps
   ```

   You will notice that each service now uses the new version of the Integration Service.

Changing the Integration Service Password

To change the password for the Integration Service API and database communications:

1. Navigate to Integration Service system or use SSH to access the server.
2. Run the following command as root:

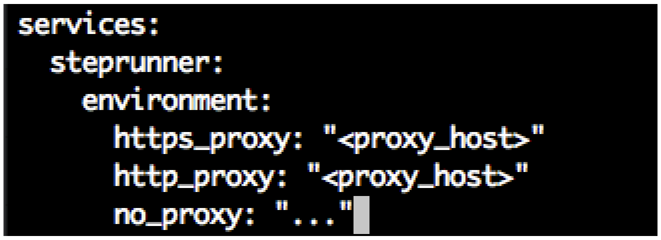
   ```
   /opt/iservices/scripts/ispasswd
   ```

3. Follow the prompts to reset the iservices password. The password must be 6 characters or more and cannot be the same as the old password.

Configuring a Proxy Server

To configure the Integration Service to use a proxy server:

1. Either go to the console of the Integration Service system or use SSH to access the Integration Service server.
2. Log in as ***isadmin*** with the appropriate password.
3. Using a text editor like "vi", edit the file **/opt/iservices/scripts/docker-compose-override.yml**.
4. In the "environment" section of the steprunner service, add the following lines:

   ```
   services:
     steprunner:
       environment:
         https_proxy: "<proxy_host>"
         http_proxy: "<proxy_host>"
         no_proxy: "..."
   ```

5. Save the settings in the file and then run the script **/opt/iservices/compose_override.sh**.

> **NOTE:** This script validates the syntax of your settings changes. If the settings are correct, the script applies the settings to your existing compose file.

6. rm and re-deploy the steprunners to use this change by typing the following commands:

```
docker service rm iservices_steprunner
docker stack deploy -c /opt/iservices/scripts/docker-compose.yml iservices
```

Configuring Security Settings

This topic explains how to change the HTTPS certificate used by the Integration Service, and it also describes password and encryption key security.

Changing the HTTPS Certificate

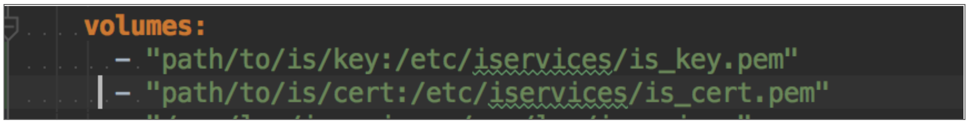
The Integration Service API and user interface only accept communications over HTTPS. By default, HTTPS is configured using an internal, self-signed certificate.

You can specify the HTTPS certificate to use in your environment by mounting the following two files in the API and user interface containers:

- /etc/iservices/is_key.pem
- /etc/iservices/is_cert.pem

To specify the HTTPS certificate to use in your environment:

1. Copy over the key and certificate to the Integration Service host.
2. Modify the /opt/iservices/scripts/docker-compose-override.yml file and mount a volume to the "gui" and "contentapi" services. The following image contains an example of the volume specification:



```
volumes:
  - "path/to/is/key:/etc/iservices/is_key.pem"
  - "path/to/is/cert:/etc/iservices/is_cert.pem"
```

3. Run the following script to validate and apply the change:

```
/opt/iservices/scripts/compose_override.sh
```

4. Re-deploy the gui service by running the following commands:

```
docker service rm iservices_gui
docker service rm iservices_contentapi
/opt/iservices/scripts/pull_start_iservices.sh
```

Using Password and Encryption Key Security

During platform installation, you can specify an Integration Service root password. This root password is also the default isadmin password.

- The root/admin password is saved in a root read-only file here: /etc/iservices/is_pass
- A backup password file is also saved in a root read-only file here: /opt/iservices/backup/is_pass

The user-created root password is also be the default Integration Service password for couchbase (:8091) and all API communications. The Integration Service platform generates a unique encryption key for every platform installation.

- The encryption key exists in a root read-only file here: /etc/iservices/encryption_key
- A backup encryption key file is also saved in a root read-only file here: /opt/iservices/backup/encryption_key

You can use the encryption key to encrypt all internal passwords and user-specified data. You can encrypt any value in a configuration by specifying `"encrypted": true`, when you POST that configuration setting to the API. There is also an option in the Integration Service user interface to select *encrypted*. Encrypted values use the same randomly-generated encryption key.

User-created passwords and encryption keys are securely exposed in the Docker containers using Docker secrets at https://docs.docker.com/engine/swarm/secrets/ to ensure secure handling of information between containers.

> **NOTE**: The encryption key must be identical between two Integration Service systems if you plan to migrate from one to another. The encryption key must be identical between High Availability or Disaster Recovery systems as well.

Helpful Docker Commands

- `docker service ls`. View available services running on the system.
- `docker service ps iservices`. View process status of all services.
- `docker service logs <service-name>`. View logs of a particular service. For example:
  - `docker service logs iservices_couchbase`
  - `docker service logs iservices_steprunner`
- `docker service scale iservices_steprunner=10`. Dynamically scale for more workers.
- `docker stack rm iservices`. Completely remove the services from running.
- `docker stack deploy -c <compose-file> iservices`. Deploy services from a defined Docker compose file.

Monitoring the Integration Service

You can use a number of ScienceLogic PowerPacks to help you monitor the health of your Integration Service system. This section describes those PowerPacks and additional resources and procedures you can use to monitor the components of the Integration Service.

PowerPacks for Monitoring Elements of the Integration Service

You can download the following PowerPacks from the ScienceLogic Customer Portal to help you monitor your Integration Service system:

- ***ScienceLogic: Integration Service PowerPack***. This PowerPack monitors the state of tasks running on the Integration Service. This PowerPack alerts users on SL1 if an application on the Integration Service fails. This PowerPack lets you monitor the status of your integration applications, and based on the events generated by this PowerPack, you can diagnose why applications failed on the Integration Service.
- ***Docker PowerPack***: This PowerPack monitors the Docker containers, services, and Swarm that manages the Integration Service containers. This PowerPack also monitors the Integration Service when it is configured for High Availability. Use version 103 or later of the Docker PowerPack to monitor Integration Service services in SL1.
- ***CouchbasePowerPack***: This PowerPack monitors the Couchbase database that the Integration Service uses for storing the cache and various configuration and application data. This data provides insight into the health of the databases and the Couchbase servers.
- ***ServiceNow Base Pack PowerPack***: This PowerPack queries ServiceNow using REST to get information about a ServiceNow instance. This PowerPack contains Run Book Automations for the Incident module integration and two Dynamic Applications that monitor the Incident and CMDB ServiceNow tables. The "ServiceNow: Incident Metrics" Dynamic Application retrieves information about incident types, priorities, and states. The "ServiceNow CMDB Configuration" Dynamic Application gathers data about all ServiceNow Configuration Items (CIs) in the CMDB that are being synced by the Integration Service, including an overall count. This CI Count value is used for billing purposes. Additionally, the "ServiceNow Open Incidents" dashboard displays a view of the information in the ServiceNow Incident table.

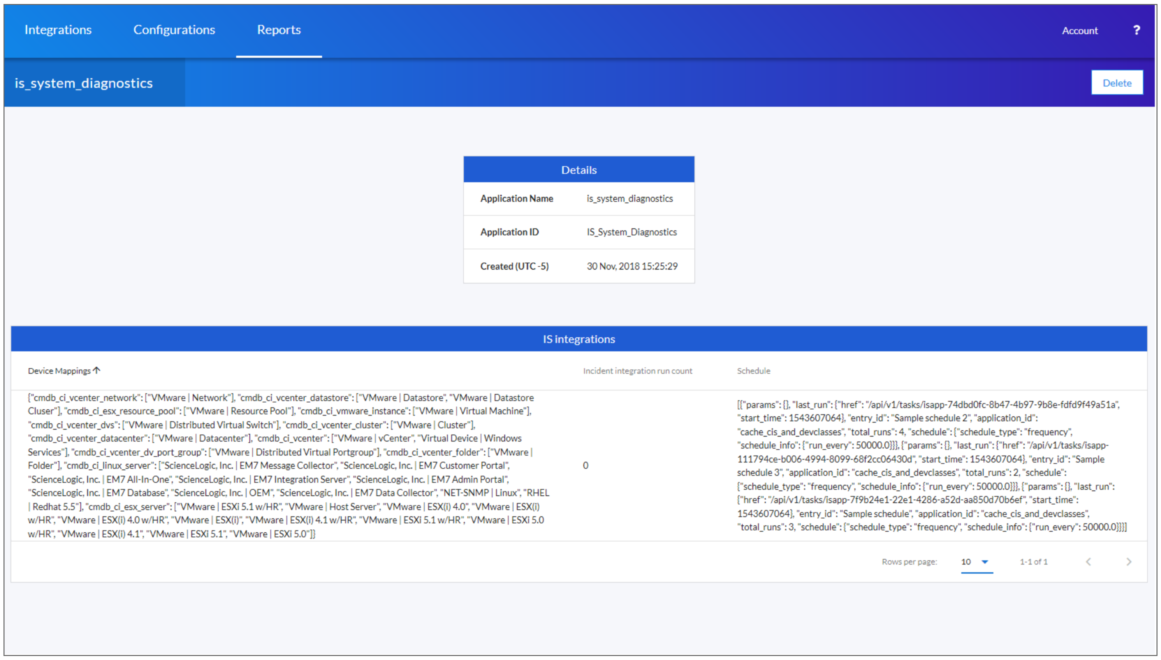
For more information about installing and using the *ScienceLogic: Integration Service*PowerPack, see ***Using SL1 to Monitor the Integration Service***.

For more information about installing and using the *ServiceNow Base Pack* PowerPack, see the ***Monitoring ServiceNow*** manual.

Integration Service System Diagnostics

The "IS System Diagnostics" integration application lets you view platform diagnostics for the Integration Service. You can use the information displayed in these diagnostics to help you troubleshoot issues with the different tools used by the Integration Service.

Running the "IS System Diagnostics" integration application in the Integration Service user interface generates a report that you can access on the **[Reports]** tab:



Details	
Application Name	is_system_diagnostics
Application ID	IS_System_Diagnostics
Created (UTC -5)	30 Nov, 2018 15:25:29

IS integrations

Device Mappings ↑	Incident integration run count	Schedule
{"cmdb_ci_vcenter_network": ["VMware | Network"], "cmdb_ci_vcenter_datastore": ["VMware | Datastore", "VMware | Datastore Cluser"], "cmdb_ci_esx_resource_pool": ["VMware | Resource Pool"], "cmdb_ci_vmware_instance": ["VMware | Virtual Machine"], "cmdb_ci_vcenter_dvs": ["VMware | Distributed Virtual Switch"], "cmdb_ci_vcenter_cluster": ["VMware | Cluster"], "cmdb_ci_vcenter_datacenter": ["VMware | Datacenter"], "cmdb_ci_vcenter": ["VMware | vCenter", "Virtual Device | Windows Services"], "cmdb_ci_vcenter_dv_port_group": ["VMware | Distributed Virtual Portgroup"], "cmdb_ci_vcenter_folder": ["VMware | Folder"], "cmdb_ci_linux_server": ["ScienceLogic, Inc. | EM7 Message Collector", "ScienceLogic, Inc. | EM7 Customer Portal", "ScienceLogic, Inc. | EM7 All-In-One", "ScienceLogic, Inc. | EM7 Integration Server", "ScienceLogic, Inc. | EM7 Admin Portal", "ScienceLogic, Inc. | EM7 Database", "ScienceLogic, Inc. | OEM", "ScienceLogic, Inc. | EM7 Data Collector", "NET-SNMP | Linux", "RHEL | Redhat 5.5"], "cmdb_ci_esx_server": ["VMware | ESXi 5.1 w/HR", "VMware | Host Server", "VMware | ESX(i) 4.0", "VMware | ESX(i) w/HR", "VMware | ESX(i) 4.0 w/HR", "VMware | ESX(i)", "VMware | ESX(i) 4.1 w/HR", "VMware | ESXi 5.1 w/HR", "VMware | ESXi 5.0 w/HR", "VMware | ESX(i) 4.1", "VMware | ESXi 5.1", "VMware | ESXi 5.0"]}	0	[{"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-74dbd0fc-8b47-4b97-9b8e-fdfd9f49a51a", "start_time": 1543607064}, "entry_id": "Sample schedule 2", "application_id": "cache_cis_and_devclasses", "total_runs": 4, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}, {"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-111794ce-b006-4994-8099-68f2cc06430d", "start_time": 1543607064}, "entry_id": "Sample schedule 3", "application_id": "cache_cis_and_devclasses", "total_runs": 2, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}, {"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-7f9b24e1-22e1-4286-a52d-aa850d70b6ef", "start_time": 1543607064}, "entry_id": "Sample schedule", "application_id": "cache_cis_and_devclasses", "total_runs": 3, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}]

This diagnostic report displays overall Integration Service settings, such as the Integration Service version, Docker version, kernel version, hostname, cluster settings, scheduled applications, CPU and memory statistics, installation date, and cache information.

TIP: If you are using a specific integration application that you want to monitor with the "IS System Diagnostics" integration application, click the **[Configure]** button and type the name of that integration application in the ***incident_create_app*** field of the **Configuration** pane.

Integration Service Endpoints

This section provides additional technical details about monitoring the Integration Service. The following information is also available in the PowerPacks listed above.

Flower API

The following Flower API endpoints return data about the Flower tasks, queues, and workers. The **tasks** endpoint returns data about task status, runtime, exceptions, and application names. You can filter this endpoint to retrieve a subset of information, and you can combine filters to return a more specific data set.

/flower/api/tasks. Retrieve a list of all tasks.

/flower/api/tasks?app_id={app_id}. Retrieve a list of tasks filtered by app_id.

/flower/api/tasks?app_name={app_name}. Retrieve a list of tasks filtered by app_name.

/flower/api/tasks?started_start=1539808543&started_end=1539808544. Retrieve a list of all tasks received within a time range.

/flower/api/tasks?state=FAILURE|SUCCESS. Retrieve a list of tasks filtered by state.

/flower/api/workers. Retrieve a list of all queues and workers

To view this information in the Flower user interface navigate to `<hostname_of_integration_service_system>/flower`.

For more information, see the Flower API Reference.

NOTE: If you use the *ScienceLogic: Integration Service* PowerPack to collect this task information, the PowerPack will create events in SL1 if a Flower task fails.

Couchbase API

The following Couchbase API endpoints return data about the Couchbase service. The **pools** endpoint represents the Couchbase cluster. In the case of the Integration Service, each ***node*** is a Docker service, and ***buckets*** represent the document-based data containers. These endpoints return configuration and statistical data about each of their corresponding Couchbase components.

<hostname_of_integration_service_system>**:8091/pools/default**. Retrieve a list of pools and nodes.

<hostname_of_integration_service_system>**:8091/pools/default/buckets**. Retrieve a list of buckets.

To view this information in the Couchbase Administrator user interface, navigate to `<hostname_of_integration_service_system>:8091`.

For more information, see the Couchbase API Reference.

NOTE: You can also use the *Couchbase* PowerPack to collect this information.

Docker Statistics

You can collect Docker information by using SSH to connect to the Docker socket. You cannot currently retrieve Docker information by using the API.

To collect Docker statistics:

1. Use SSH to connect to the Integration Service instance.
2. Run the following command:

```
curl --unix-socket /var/run/docker.sock http://docker<PATH>
```

where `<PATH>` is one of the following values:

- `/info`
- `/containers/json`
- `/images/json`
- `/swarm`
- `/nodes`
- `/tasks`
- `/services`

NOTE: You can also use the *Docker* PowerPack to collect this information.

Integration Service Log Files

Use the following procedures to help you locate and understand the contents of the various log files related to the Integration Service.

Accessing Docker Log Files

The Docker log files contain information logged by all containers participating in the Integration Service. The information below is also available in the PowerPacks listed above.

To access Docker log files:

1. Use SSH to connect to the Integration Service instance.
2. Run the following Docker command:

```
docker service ls
```

3. Note the Docker service name, which you will use for the *<service_name>* value in the next step.
4. Run the following Docker command:

```
docker service logs -f <service_name>
```

Accessing Local File System Logs

The local file system logs display the same information as the Docker log files. These log files include debug information for all of the Integration Service integration applications and all of the Celery worker nodes.

To access local file system logs:

1. Use SSH to connect to the Integration Service instance.
2. Navigate to the **/var/log/iservices** directory to view the log files.

Understanding the Contents of Log Files

The pattern of deciphering log messages applies to both Docker logs and local log files, as these logs display the same information.

The following is an example of a message in a Docker log or a local file system log:

```
"2018-11-05 19:02:28,312","FLOW","12","device_sync_sciencelogic_to_servicenow","ipaas_
logger","142","stop Query and Cache ServiceNow CIs|41.4114570618"
```

You can parse this data in the following manner:

```
'YYYY-MM-DD' 'HH-MM-SS,ss' 'log-level' 'process_id' 'is_app_name' 'file' 'lineOfCode'
'message'
```

To extract the runtime for each individual task, use regex to match on a log line. For instance, in the above example, there is the following sub-string:

```
"stop Query and Cache ServiceNow CIs|41.4114570618"
```

Use regex to parse the string above:

```
"stop …… | …"
```

where:

- Everything after the `|` is the time taken for execution.
- The string between `"stop"` and `|` represents the step that was executed.

In the example message, the "Query and Cache ServiceNow CIs" step took around 41 seconds to run.

Chapter 3

Configuring the Integration Service for High Availability

Overview

This chapter describes how to create High Availability configurations to protect the data in the Integration Service. This chapter covers the following topics:

Configuring a High Availability Environment for the Integration Service

Because the Integration Service uses the Docker Swarm tool to maintain its cluster and automatically re-balance services across nodes, ScienceLogic strongly recommends that you implement the following best practices from Docker, Couchbase, and RabbitMQ. The topics in this section describe those best practices, along with requirements and frequently asked questions.

NOTE: To support automatic failover of the Couchbase database without manual intervention, you must set up at least *three* nodes. You can achieve High Availability with two nodes, and no data will be lost in the event of a single node failure. However, with only two nodes, automatic failover will not function and you must manually perform the failover steps.

Docker Swarm Requirements for High Availability

After implementing Docker Swarm High Availability, if a node goes down, all the services on that failed node can be dynamically re-provisioned and orchestrated among the other nodes in the cluster. High Availability for Swarm also facilitates network connections with the various other High Availability components.

Docker Swarm requires the following:

- The cluster contains at least three nodes running as managers. With three nodes, there can be a quorum vote between managers when a node is failed over.
- A load balancer with a virtual IP running in front of all nodes in the cluster. The load balancer allows user interface requests to be distributed among each of the hosts in the case one of the hosts fails.

For more information, see the Docker High Availability Documentation.

Docker Swarm Frequently Asked Questions for High Availability

What happens if I only use two nodes and one node fails?

Using only two nodes does not meet Docker's High Availability requirements, so automatic High Availability and failover cannot be guaranteed. In the event of a failure of one out of two nodes, depending on which services fail, the Integration Service system might not be functional until a user logs in and performs some manual actions, such as removing the other failed node from the cluster.

After you perform these manual failover actions, the Integration Service will be back up and running.

What happens if I use three nodes and two of the nodes fail?

Docker fault tolerance is limited to one failure in a three-node cluster. If more than one node goes down in a three-node cluster, automatic High Availability and failover cannot be guaranteed, and manual intervention may be required. Adding more nodes is the only way to increase the fault tolerance.

In the event of a two out of three failure, after you perform manual failover actions, the Integration Service system will be back up and running.

For more information about the manual failover steps, see the ***Failover*** section.

Couchbase Database Requirements for High Availability

Couchbase High Availability ensures that no integration application, configuration, or step data will be lost in the event of a node failure. To support automatic failover, Couchbase requires at least *three* nodes in the high availability cluster.

Each node will have an independent and persistent storage volume that is replicated throughout the cluster. Alternatively, shared storage can be used instead of independent persistent volumes. This replication ensures that data is replicated in all places, and if a single node goes down, no data will be lost.

For more information, see the Couchbase documentation.

Couchbase Database Frequently Asked Questions for High Availability

What if I don't have three nodes? If I only use two nodes, what happens during a failure?

In the event of a failure of one out of two nodes, no data will be lost, because the data is being replicated. With only two nodes, automatic failover will not function, and you will need to perform manual failover actions. For more information about the manual failover steps, see the ***Failover*** section.

What if I have three nodes and two of them fail?

In the event of a failure of two out of three nodes, no data will be lost, because the data is being replicated.

If multiple Couchbase data nodes go down at the same time, automatic failover might not occur (not even nodes for quorum to failover). You will then need to perform manual failover steps. After you perform these manual actions, the Integration Service will be operational again. For more information about the manual failover steps, see the ***Failover*** section.

> **NOTE**: If you have three nodes, automatic failover is supported by Docker Swarm and Couchbase. If you have less than three nodes, follow the steps in the ***Failover*** section to manually fail over a system to regain Integration Service functionality.

RabbitMQ Clustering and Persistence for High Availability

Implementing RabbitMQ High Availability ensures that if any integrations or tasks are waiting in the Rabbit queue, those tasks will not be lost if a node containing the Rabbit queue fails.

> **NOTE**: You can switch between both single-node and cluster options at any time during deployment.

RabbitMQ clustering requires a Docker Swarm configuration with multiple nodes. For more information, see *Configuring Docker Swarm*.

As a best practice for security, enable the user interface only temporarily during cluster configuration. The default user interface login is *guest/guest*.

RabbitMQ Option 1: Persisting Queue to Disk on a Single Node (Default Configuration)

With this configuration, the Integration Service queue runs on a single node, and the queue is persisted on disk. As a result, if the Integration Service stack is removed and re-deployed, no messages are lost during the downtime. Any messages that exist in the queue before the stack is stopped continue to exist after the stack is re-deployed.

Potential Risks and Mitigations

Because the queue runs on a single node, if that node fails, then the queue and its related data might be lost.

You can mitigate data loss by persisting the queues on your choice of network shared storage, so that if the queue fails on one node, the queue and its messages can be brought back up on another node.

Requirements/Setup (Enabled by Default)

- You must define a static hostname for the RabbitMQ host in the docker-compose file. The default is *rabbit_node1.isnet*.
- You must mount a volume to /var/lib/rabbitmq in the docker-compose file.

Example Compose Definition

```
rabbitmq:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node1.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"
  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node1.isnet
```

RabbitMQ Option 2: Clustering Nodes with Persistent Queues on Each Node

This configuration lets multiple nodes join a RabbitMQ cluster. When you include multiple nodes int he RabbitMQ cluster, all queue data, messages, and other necessary information is automatically replicated and persisted on all nodes in the cluster. If any node fails, then the remaining nodes in the cluster continue maintaining and processing the queue.

Because the RabbitMQ cluster includes disk-persisted queues, if all nodes in the Rabbit cluster fail, or if the service is removed entirely, then no data loss should occur. Upon restart, the nodes will resume with the same cluster configuration and with the previously saved data.

If you include multiple nodes in a RabbitMQ cluster, the Integration Service automatically applies an HA policy of all-node replication, with retroactive queue synchronization disabled. For more information, refer to the RabbitMQ documentation.

Potential Risks and Mitigations

If you create a Docker Swarm cluster with only two nodes, the cluster might stop functioning if a single node fails. To prevent this situation, include at least *three* nodes in each cluster.

Requirements/Setup

For a Docker Swarm configuration with multiple independent nodes:

- Both RabbitMQ services must be "pinned" to each of the two nodes. See the **Example Compose Definition** below.
- You must add a new RabbitMQ service to the docker-compose.yml file. This new service should have a hostname and alias following the designated pattern. The designated pattern is: *rabbit_nodex.isnet*, where *x* is the node number. This configuration supports up to 20 clustered nodes by default.
- After you update the docker-compose.yml file, the nodes will auto-cluster when you perform a deployment.

Example Compose Definition of Two Clustered Rabbit Services

```
rabbitmq:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node1.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"
  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node1.isnet
  deploy:
    placement:
      constraints:
        - node.hostname == node-number-1.domain
rabbitmq2:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node2.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"

  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node2.isnet
  deploy:
    placement:
        constraints:
          - node.hostname == node-number-2.domain
```

Checking the Status of a RabbitMQ Cluster

This section contains commands and additional resources for administering your clusters.

To check the status of your clustered RabbitMQ environment:

1. Run `docker ps` and locate the iservices_rabbit container.
2. Run the following command on the RabbitMQ container:

   ```
   docker exec -it [container_id] /bin/bash
   ```

You can run the following commands for more information:

- `rabbitmqctl cluster_status`. Returns information about the current cluster status, including nodes in the cluster, and failed nodes.
- `rabbitmqctl list_policies`. Returns information about current policies. Ensure that the ha-all policy is automatically set for your cluster.

For additional cluster-related administrative commands, see the RabbitMQ Cluster Management documentation page.

System Requirements for High Availability

The High Availability solution has the following system requirements:

1. Ensure that your Integration Service system has been updated with `yum upgrade`.
2. Run the following commands to open up the proper firewall ports for Docker Swarm on each swarm node:

   ```
   firewall-cmd --add-service docker-swarm --permanent
   firewall-cmd --reload
   ```

3. Ensure that /etc/iservices/is_pass and /etc/iservices/encryption_key are identical on all clustered nodes.
4. Ensure that NTP is properly configured on all nodes:

   - Run the following command:

     ```
     edit /etc/chronyd.conf
     ```

   - Add NTP servers. If you want to use the pool.ntp.org NTP servers, remove the **.ol.** from the domain names.
   - Enable chronyd by running the following commands:

     ```
     systemctl start chronyd
     systemctl enable chronyd
     timedatectl #ensure ntp is enabled is yes and ntp sync is yes
     ```

Configuring Clustering and High Availability

This section describes how to configure clustering and High Availability with Docker Swarm and the Couchbase database, using either two nodes or three or more nodes.

NOTE: This topic assumes you are using Integration Service ISOs for each node, which includes an initial Docker Swarm node configuration. The use of the Integration Service ISO is not required, however. You could instead deploy another node (without using the Integration Service ISO) and configure a Linux operating system based on Red Hat. You could then add that system to the swarm.

Configuring Docker Swarm

To configure Docker Swarm for clustering (two nodes or three or more nodes) and High Availability:

1. If you do not already have Integration Service running in your environment, install the Integration Service on a single node. Doing this creates a single-node Docker Swarm manager. For more information, see ***Installing the Integration Service***.
2. Ensure that NTP is configured on all swarm nodes. For more information, see ***System Requirements for High Availability***.
3. SSH to the Docker Swarm manager and run the following command to retrieve the join token. Make note of the token, because you need it to join a node to the swarm:

   ```
   docker swarm join-token manager
   ```

4. Run the following commands on each Docker Swarm manager that you want to join to the cluster:

   ```
   docker swarm init
   docker swarm join --token <join token> <swarm manager ip>:<port>
   ```

5. Run the following command to verify that the nodes have been added:

   ```
   docker node ls
   ```

6. If you are using local images and not connecting to Docker Hub, load docker images on the other swarm nodes:

   ```
   docker load -i /opt/iservices/images/is-api:1.x.x.tar
   docker load -i /opt/iservices/images/is-couchbase:1.x.x.tar
   docker load -i /opt/iservices/images/is-gui:1.x.x.tar
   docker load -i /opt/iservices/images/is-worker:1.x.x.tar
   docker load -i /opt/iservices/images/is-rabbit:3.x.x-1.tar
   docker load -i /opt/iservices/images/redis:4.x.x.tar
   docker load -i /opt/iservices/images/visualizer.tar
   ```

   where:

- *x.x* is the version number of the .tar file.

Configuring the Couchbase Database

In a Couchbase cluster you have a master and one or more worker nodes. At least one worker node is required for a Couchbase cluster.

To add a Couchbase worker node:

1. Add the following line to constrain the Couchbase container to a single Docker Swarm node at the bottom of the **Couchbase** section:

```
deploy:
...
  hostname: couchbase.isnet
  deploy:
    placement:
      constraints:
        - node.hostname == <name of Docker Swarm node>

  networks:
    isnet:
      aliases:
        - couchbase
        - couchbase.isnet

  environment:
    db_host: couchbase.isnet
```

2. Add the couchbase-worker and couchbase-worker2 section. deploy > replicas on the workers should be set to 0:

```
couchbase-worker:
  image: repository.auto.sciencelogic.local:5000/is-couchbase:feature-INT-1208-HA-
  IS-Services
  container_name: couchbase-worker.isnet
  volumes:
    - "/var/data/couchbase:/opt/couchbase/var"
  deploy:
    placement:
      constraints:
        - node.hostname == <name of Docker Swarm node>
  networks:
    isnet:
      aliases:
        - couchbase-worker
        - couchbase-worker.isnet

  hostname: couchbase-worker.isnet
  ports:
    - "8095:8091"
  secrets:
    - is_pass
    - encryption_key
  ulimits:
    nofile: 80000
    core: 100000000
    memlock: 100000000
  environment:
    TYPE: 'WORKER'
    AUTO_REBALANCE: 'true'
    db_host: 'couchbase'
  depends_on:
    - couchbase
```

NOTE: This deployment makes the Couchbase worker user interface available on port 8095 of the Docker Swarm stack. If the master node goes down, or if the primary Couchbase user interface is not available on port 8091, you can still access the secondary Couchbase user interface through port 8095.

3. Add couchbase-worker to the db_host setting for contentapi:

```
contentapi:
...
  environment:
  ...
    db_host: 'couchbase,couchbase-worker,couchbase-worker2'
```

4. All db_host vars in docker-compose should be in the following format:

```
db_host: 'couchbase,couchbase-worker,couchbase-worker2'
```

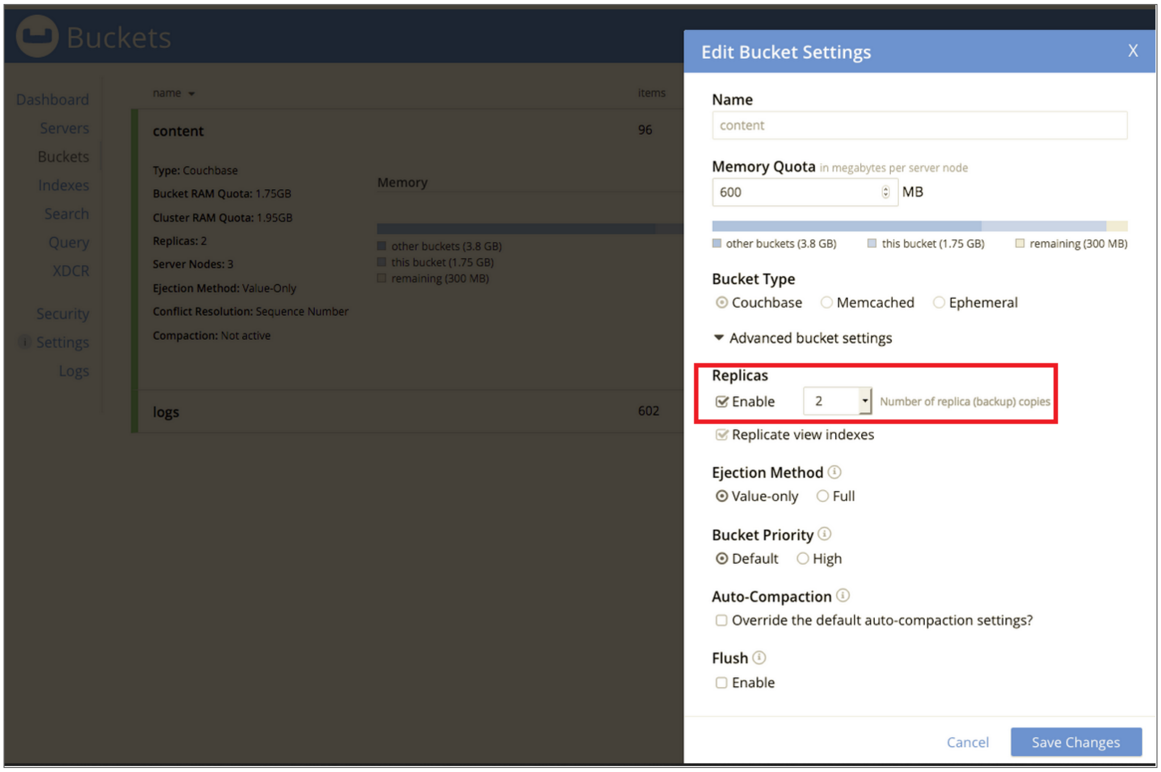
5. If using the override file, run the **compose-override.sh** script to generate the docker-compose.yml file.
6. Deploy the stack with only the couchbase node by editing the replicas on couchbase-worker to 1 and running the following command:

```
docker stack deploy -c <location of compose file> iservices
```

7. After the two-node Couchbase cluster has been successfully deployed and the secondary indexes are successfully added, edit the replicas on couchbase-worker2 to 1 and run the following command:

```
docker stack deploy -c <location of compose file> iservices
```

8. Set the replicas in the docker-compose-override.yml file as well.
9. After the second worker is added, set the number of replicas to "2" on each bucket (content and logs) in the Couchbase Administrator user interface and click **[Save Changes]**:



content 96

Type: Couchbase
Bucket RAM Quota: 1.75GB
Cluster RAM Quota: 1.95GB
Replicas: 2
Server Nodes: 3
Ejection Method: Value-Only
Conflict Resolution: Sequence Number
Compaction: Not active

other buckets (3.8 GB)
this bucket (1.75 GB)
remaining (300 MB)

logs 602

Name: content

Memory Quota in megabytes per server node: 600 MB

other buckets (3.8 GB) this bucket (1.75 GB) remaining (300 MB)

Bucket Type
● Couchbase ○ Memcached ○ Ephemeral

Advanced bucket settings

Replicas
☑ Enable 2 Number of replica (backup) copies

☑ Replicate view indexes

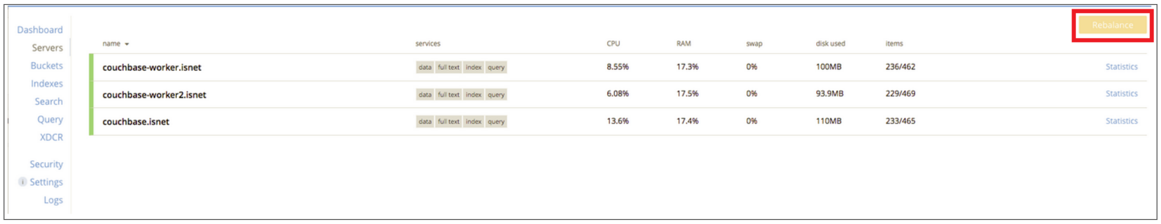
Ejection Method
● Value-only ○ Full

Bucket Priority
● Default ○ High

Auto-Compaction
□ Override the default auto-compaction settings?

Flush
□ Enable

10. Rebalance the cluster by navigating to the **Servers** section of the Couchbase Administrator user interface and clicking the **Rebalance** button:



name	services	CPU	RAM	swap	disk used	items	
couchbase-worker.isnet	data full text index query	8.55%	17.3%	0%	100MB	236/462	Statistics
couchbase-worker2.isnet	data full text index query	6.08%	17.5%	0%	93.9MB	229/469	Statistics
couchbase.isnet	data full text index query	13.6%	17.4%	0%	110MB	233/465	Statistics

Code Example: docker-compose-override.yml

The following section includes a complete example of the **/opt/iservices/scripts/docker-compose-override.yml** file for a three-node Couchbase and RabbitMQ clustered deployment:

NOTE: If shared volumes are available in the cluster, the deploy placement can be omitted and removed.

```
version: '3.2'
services:
  steprunner:
```

```
  environment:
    db_host: couchbase.isnet,couchbase-worker2.isnet,couchbase-worker.isnet

scheduler:
  environment:
    db_host: couchbase.isnet,couchbase-worker2.isnet,couchbase-worker.isnet

couchbase:
  environment:
    db_host: 'couchbase.isnet'
  deploy:
    placement:
      constraints:
        - node.hostname == <swarm node hostname>
  networks:
    isnet:
      aliases:
        - couchbase
        - couchbase.isnet
  hostname: couchbase.isnet

couchbase-worker:
  image: sciencelogic/is-couchbase:1.7.0
  container_name: couchbase-worker
  volumes:
    - "/var/data/couchbase:/opt/couchbase/var"
  ports:
    - "8100:8091"
  deploy:
    placement:
      constraints:
        - node.hostname == <swarm node hostname>
  networks:
    isnet:
      aliases:
        - couchbase-worker
        - couchbase-worker.isnet
  hostname: couchbase-worker.isnet
  secrets:
    - is_pass
    - encryption_key
  environment:
    TYPE: 'WORKER'
    AUTO_REBALANCE: 'true'
    db_host: 'couchbase'
  depends_on:
    - couchbase

couchbase-worker2:
  image: sciencelogic/is-couchbase:1.7.0
  container_name: couchbase-worker2
  ports:
    - "8100:8091"
  volumes:
    - "/var/data/couchbase:/opt/couchbase/var"
  deploy:
    replicas: 0
    placement:
      constraints:
        - node.hostname == <swarm node hostname>
  networks:
    isnet:
      aliases:
        - couchbase-worker2
        - couchbase-worker2.isnet
  hostname: couchbase-worker2.isnet
  secrets:
    - is_pass
    - encryption_key
  environment:
    TYPE: 'WORKER'
    AUTO_REBALANCE: 'true'
    db_host: 'couchbase'
  depends_on:
    - couchbase

rabbitmq:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node1.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"
  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node1.isnet
    deploy:
      placement:
        constraints:
          - node.hostname == <swarm node hostname>

rabbitmq2:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node2.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"
  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node2.isnet
    deploy:
      placement:
        constraints:
          - node.hostname == <swarm node hostname>

rabbitmq3:
  image: sciencelogic/is-rabbit:3.7.7-1
  hostname: rabbit_node3.isnet
  volumes:
    - "rabbitdb:/var/lib/rabbitmq"
  networks:
    isnet:
      aliases:
        - rabbit
        - rabbit_node3.isnet
    deploy:
      placement:
        constraints:
          - node.hostname == <swarm node hostname>
contentapi:
  environment:
      db_host: 'couchbase.isnet,couchbase-worker.isnet,couchbase-worker2.isnet'

volumes:
rabbitdb2:
rabbitdb3:
```

Scale iservices-contentapi

To scale out the iservices-contentapi to "3" to distribute the service across the three nodes, run the following command:

```
docker service scale iservices-contentapi=3
```

Manual Failover

If you have a cluster with three or more nodes that is not configured with automatic failover, you must perform the following manual failover steps.

NOTE: If you can access the Couchbase Administrator user interface (**http://<*IP of Integration Service*>:8091**) on the node that is still running, you can simply click the **[Failover]** button in the Couchbase Administrator user interface instead of manually running the couchbase-cli commands below.

NOTE: In a three-node cluster, a single failed node will be automatically removed, you will still need to perform a re-balance.

To initiate a manual failover:

1. Log in to the Docker Swarm node where the node that is running resides.
2. Remove any failed managers from the cluster by running the following Docker commands:

   ```
   docker swarm init --force-new-cluster
   docker node rm <failed node id>
   ```

3. Run `docker ps` to identify the Container ID of the running Couchbase container.
4. Connect to the Docker container:

   ```
   docker exec -i -t <container id> /bin/bash
   ```

5. Identify the failed node by running the commands:

   ```
   couchbase-cli server-list -c couchbase -u isadmin -p <password>
   couchbase-cli server-list -c couchbase-worker -u isadmin -p <password>
   ```

6. One of the previous commands will show a failed node. Copy the IP address and port number of the failed node for step 7.

7. Use the currently running cluster and the failed node's IP address and port to run the following command to failover:

   ```
   couchbase-cli failover -c <couchbase|couchbase-worker> -u isadmin -p <password> --
   server-failover <ip:port> --force
   ```

   For example, if the functioning node is *couchbase-worker*, and the ip:port of the failed service is *10.0.0.4:4379*, then the command would be:

   ```
   couchbase-cli failover -c couchbase-worker -u isadmin -p <password> --server-
   failover 10.0.0.4:4379 --force
   ```

8. Rebalance the cluster using the functioning container name:

   ```
   couchbase-cli rebalance -c <cluster|cluster-worker> -u isadmin -p <password>
   ```

9. In the unlikely event that a failover occurs and no queries can be performed, validate that the indexes exist, and if not, rebuild them. To rebuild the primary indexes, run the following commands:

   ```
   cbq -u isadmin
   CREATE PRIMARY INDEX ON content;
   CREATE PRIMARY INDEX ON logs;
   ```

To recover a Docker Swarm node:

1. Re-deploy the node.
2. Add a new manager node to the swarm stack.

To restore the failed Couchbase node:

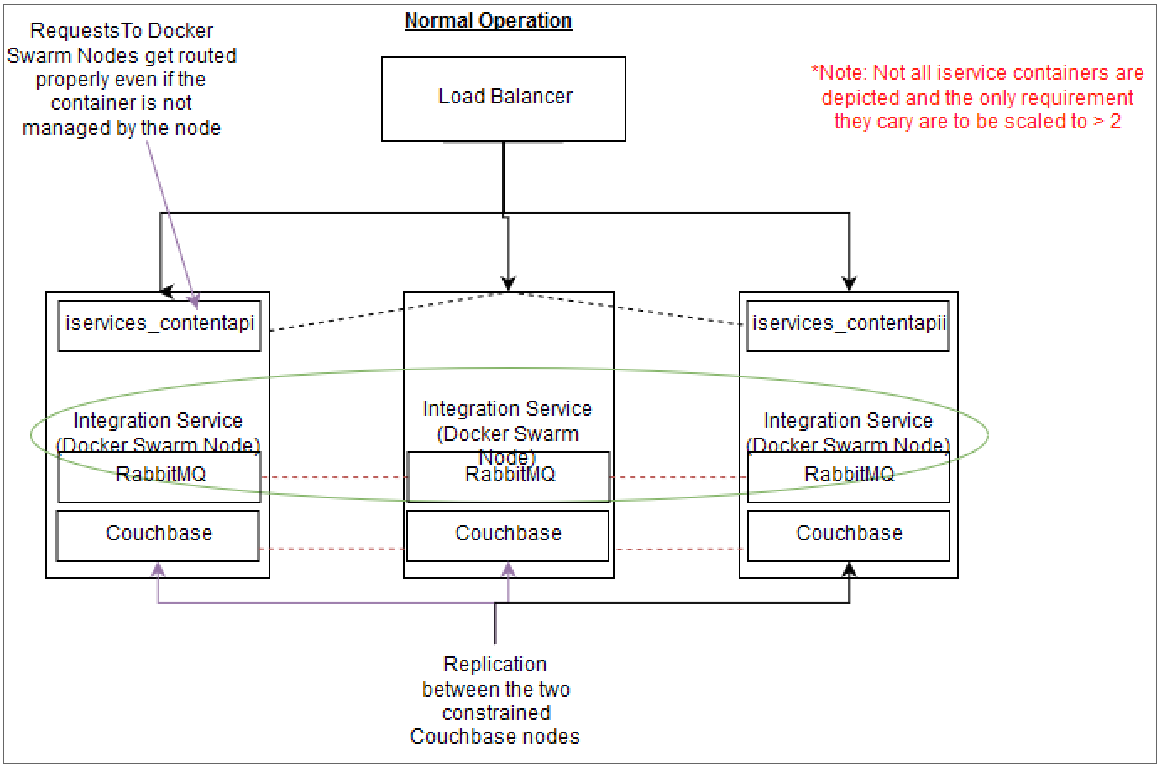
1. Log in to the node where the failed Couchbase cluster node was pinned.
2. Run *one* of the following commands, depending on the Couchbase node being recovered:
   - `docker service scale iservices_couchbase=0`
   - `docker service scale iservices_couchbase-worker=0`
3. If the Docker Swarm node was restored and not rebuilt, remove files from the old container:

   ```
   rm -rf /var/data/couchbase/*
   docker service scale iservices_couchbase scale 1
   ```

   A new node is added to Couchbase that will automatically re-balance the cluster after it is added.

Additional Configuration Information

The following diagram describes a typical High Availability configuration that uses load balancing, replication, and failover:


Normal Operation
Requests To Docker Swarm Nodes get routed properly even if the container is not managed by the node
Load Balancer
*Note: Not all iservice containers are depicted and the only requirement they cary are to be scaled to > 2
iservices_contentapi
Integration Service (Docker Swarm Node)
RabbitMQ
Couchbase
Integration Service (Docker Swarm Node)
RabbitMQ
Couchbase
iservices_contentapii
Integration Service (Docker Swarm Node)
RabbitMQ
Couchbase
Replication between the two constrained Couchbase nodes

HAProxy Configuration (Optional)

The following example configuration describes using HAProxy as a load balancer:

```
...

frontend http_front
  bind *:80
  bind *:443
  option tcplog
  mode tcp
  tcp-request inspect-delay 5s
  default_backend http_back

backend http_back
  mode tcp
  balance roundrobin
  server master1 <docker swarm node 1 ip>:443 check
  server master2 <docker swarm node 2 ip>:443 check
  server master3 <docker swarm node 3 ip>:443 check
```

Known Issues

The following section describes the known issues you might encounter with the High Availability solution and how to address those issues.

Docker Network Alias is incorrect

If you experience issues with the iservices_contentapi container, the Alias IP might be incorrect.

To address this issue, run the following commands on the relevant node:

```
docker service scale iservices_contentapi=0
docker service scale iservices_contentapi=1 (or another number as needed)
```

NOTE: This issue was addressed in the **docker-ce 18.06** version of Docker, which is required for version 1.8.0 of the Integration Service.

Docker container on last swarm node cannot communicate with other swarm nodes

This is an issue with the Encapsulating Security Payload (ESP) protocol not being enabled in firewalld. You can enable the ESP protocol with the firewalld docker-swarm script.

To address this issue, add the following firewall rule to each node:

```
firewall-cmd --add-protocol=esp --permanant
firewall-cmd --reload
```

Couchbase service does not start, remains hung at nc -z localhost

To address this issue, stop the container where this is happening and remove its persistent volume:

```
rm -rf /var/data/couchbase
```

Couchbase-worker fails to connect to master

A connection failure might happen a few times when a stack is freshly deployed. You can ignore these messages, and the worker should eventually connect to the master.

The Integration Service user interface fails to start after a manual failover of the swarm node

To address this issue, run the following commands on the relevant node:

```
docker stack rm iservices
systemctl restart docker
docker stack deploy -c docker-compose.yml iservices
```

The Integration Service user interface returns 504 errors

Ensure that your Integration Service systems have been updated with `yum upgrade`.

NTP should be used, and all node times should be in sync

If all nodes time are not in sync, you might experience issues with the iservices_steprunners.

The following is an example of a Docker Swarm error caused by the time not being in sync:

```
Error response from daemon: certificate (1 - 2v4umws4pxag6kbxaelwfl3vf) not valid
before Fri, 30 Nov 2018 13:47:00 UTC, and it is currently Fri, 30 Nov 2018 06:41:24
UTC: x509: certificate has expired or is not yet valid
```

For more information, see *System Requirements for High Availability*.

Example Logs from Flower

```
iservices_flower.1.jg6glaf298d2@is-scale-05 | [W 181023 20:17:40 state:113]
Substantial drift from celery@1ee384863e37 may mean clocks are out of sync. Current
drift is iservices_flower.1.jg6glaf298d2@is-scale-05 | 18 seconds. [orig: 2018-10-23
20:17:40.090473 recv: 2018-10-23 20:17:58.486666]
```

Chapter

4

Managing Integration Applications

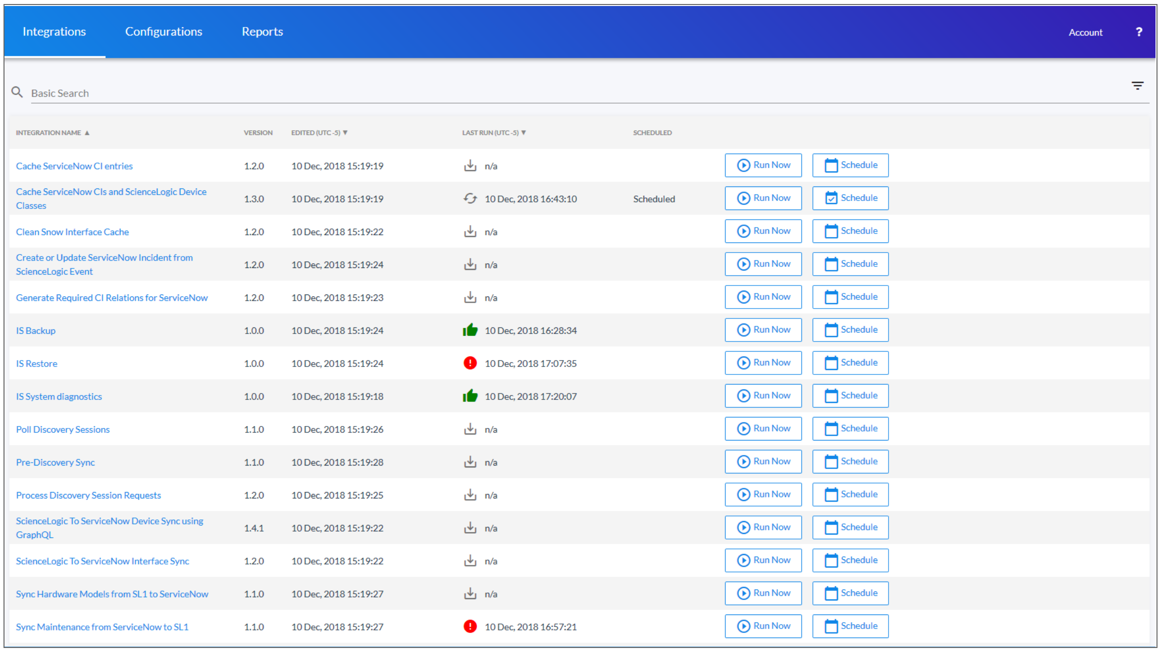
Overview

This chapter describes how to use the Integration Service Application Registry and Editor to run and schedule integrations.

This chapter covers the following topics:

Viewing the List of Integration Applications

The **Integration Application Registry** page, or the **[Integrations]** tab, allows you to view a list of integration applications on your system. From this page you can view, run, and schedule applications:



INTEGRATION NAME ▲	VERSION	EDITED (UTC -5) ▼	LAST RUN (UTC -5) ▼	SCHEDULED		
Cache ServiceNow CI entries	1.2.0	10 Dec, 2018 15:19:19	⤓ n/a		Run Now	Schedule
Cache ServiceNow CIs and ScienceLogic Device Classes	1.3.0	10 Dec, 2018 15:19:19	↻ 10 Dec, 2018 16:43:10	Scheduled	Run Now	Schedule
Clean Snow Interface Cache	1.2.0	10 Dec, 2018 15:19:22	⤓ n/a		Run Now	Schedule
Create or Update ServiceNow Incident from ScienceLogic Event	1.2.0	10 Dec, 2018 15:19:24	⤓ n/a		Run Now	Schedule
Generate Required CI Relations for ServiceNow	1.2.0	10 Dec, 2018 15:19:23	⤓ n/a		Run Now	Schedule
IS Backup	1.0.0	10 Dec, 2018 15:19:24	👍 10 Dec, 2018 16:28:34		Run Now	Schedule
IS Restore	1.0.0	10 Dec, 2018 15:19:24	❗ 10 Dec, 2018 17:07:35		Run Now	Schedule
IS System diagnostics	1.0.0	10 Dec, 2018 15:19:18	👍 10 Dec, 2018 17:20:07		Run Now	Schedule
Poll Discovery Sessions	1.1.0	10 Dec, 2018 15:19:26	⤓ n/a		Run Now	Schedule
Pre-Discovery Sync	1.1.0	10 Dec, 2018 15:19:28	⤓ n/a		Run Now	Schedule
Process Discovery Session Requests	1.2.0	10 Dec, 2018 15:19:25	⤓ n/a		Run Now	Schedule
ScienceLogic To ServiceNow Device Sync using GraphQL	1.4.1	10 Dec, 2018 15:19:22	⤓ n/a		Run Now	Schedule
ScienceLogic To ServiceNow Interface Sync	1.2.0	10 Dec, 2018 15:19:22	⤓ n/a		Run Now	Schedule
Sync Hardware Models from SL1 to ServiceNow	1.1.0	10 Dec, 2018 15:19:27	⤓ n/a		Run Now	Schedule
Sync Maintenance from ServiceNow to SL1	1.1.0	10 Dec, 2018 15:19:27	❗ 10 Dec, 2018 16:57:21		Run Now	Schedule

In the **Last Run** column, you can view the current status of an integration application:

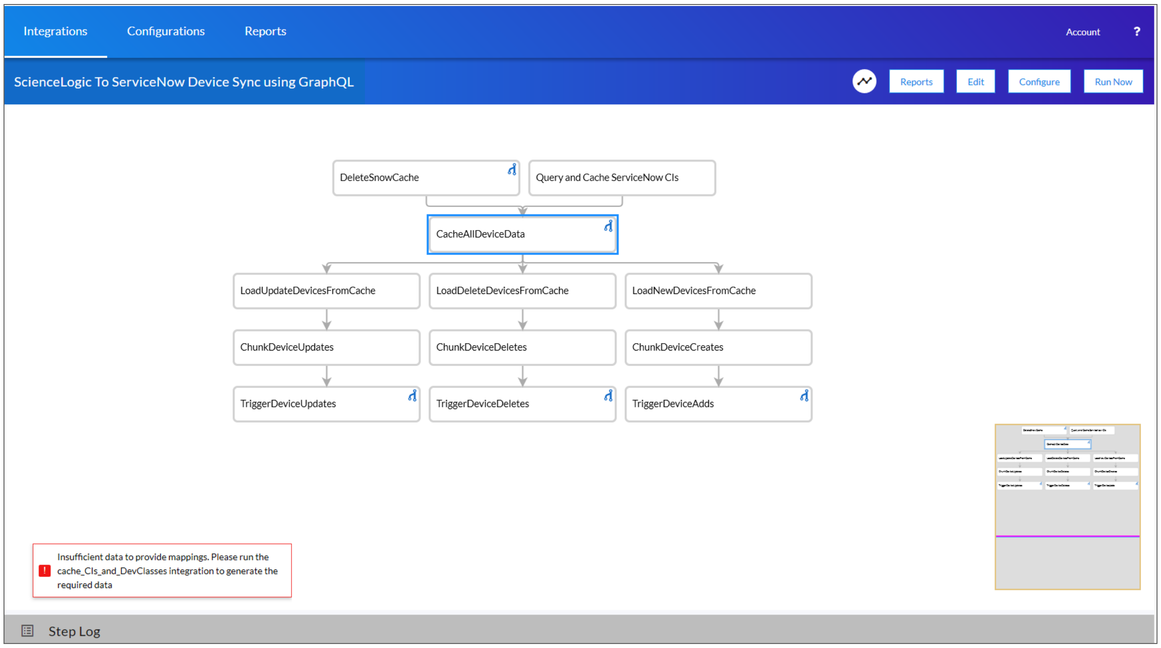
Icon	Status
👍	The integration application ran successfully.
↻	The integration application is currently running.
❗	The integration application failed to run successfully.
⤓	The integration application has not been run.

Some of the integration applications on the **[Integrations]** tab are *internal* applications that you should not run directly. Instead, other "parent" integration applications run these internal applications.

TIP: To view the internal integration applications, click the Filter icon ( ) at the top right of the **[Integrations]** tab and select *Show Hidden Integrations*. Internal integration applications are hidden by default.

Using the Integration Application Editor

When you click the name of an integration application, an **Integration Application Editor** page for that application appears:



Insufficient data to provide mappings. Please run the cache_CIs_and_DevClasses integration to generate the required data

In the main pane of the page, the steps for the application are organized as a flowchart. The arrows indicate the order in which the steps will execute when you run the application.

If a step triggers a child application, a branch icon ( ) appears in the upper right-hand corner of the step. You can double-click the branch icon to open the child application, or you can click the branch icon once to display the triggered application's run ID as a link in a pop-up window. If no run ID is present, the branch icon displays "NONE".

The buttons let you view past runs, reports for the integration application, and edit, configure, or run the integration application.

In the bottom left-hand corner of the page, you can view the status of the application. In the example above, the status is "Run: pending". Below the status is the **Step Log** pane, which displays the logs from a step you selected in the main pane.

In the bottom right-hand corner of the page is a smaller version of the application. You can click and drag on this version to move or scroll through the steps in the main pane.

Default Steps

The Integration Service system includes some already-defined steps:

- MicrosoftSqlDescribe
- MicrosoftSqlInsert
- MicrosoftSqlSelect
- MySqlDescribe
- MySqlInsert
- MySqlSelect
- QueryGQL
- QueryREST
- stepTemplate

To view the code for one of these steps:

1. Use Postman, cURL, or another REST API tool, or use the API `GET /steps` to download the steps:

   ```
   URL_for_your_Integration_Service_system/api/v1/steps/step_name
   ```

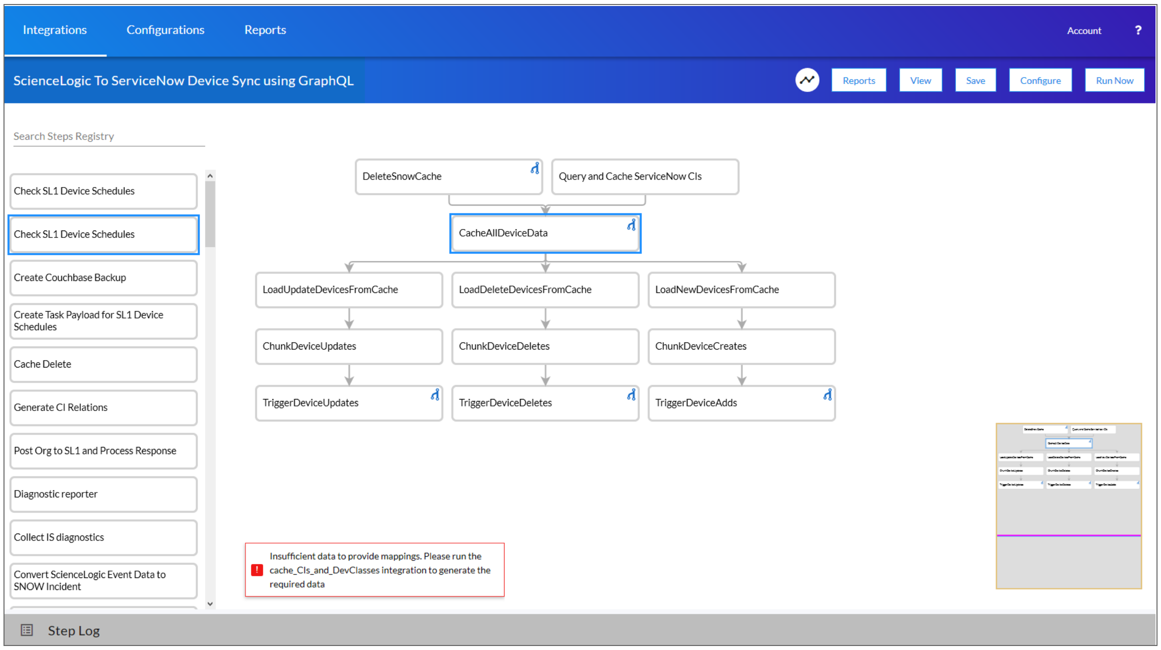
   where:

   - *URL_for_your_Integration_Service_system* is the IP address or URL for the Integration Service.
   - *step_name* is the name of the step you want to view.

Editing an Integration Application

To edit an integration application:

1. From the **Integration Application Editor** page, click the **[Edit]** button. The **Search Steps Registry** pane appears, with a list of all of the steps that are available for that integration:



Insufficient data to provide mappings. Please run the cache_CIs_and_DevClasses integration to generate the required data

2. Scroll through the **Search Steps Registry** pane or use the **Search** field at the top of the pane to find the step you want to add.
3. Click the step you want to add, drag it to the main pane of the **Integration Application Editor** page, and drop it into the integration application .
4. To adjust the position of any step in the integration application, click the step you want to move and drag it to its new location.
5. To redirect the arrows connecting the steps, click an arrow and drag it to reposition it.
6. To remove a step, click the step to select it and press the **[Delete]** key on your keyboard.
7. To save the changes you made to the integration application, click the **[Save]** button.
8. To stop editing and close the **Search Steps Registry** panel, click the **[View]** button.

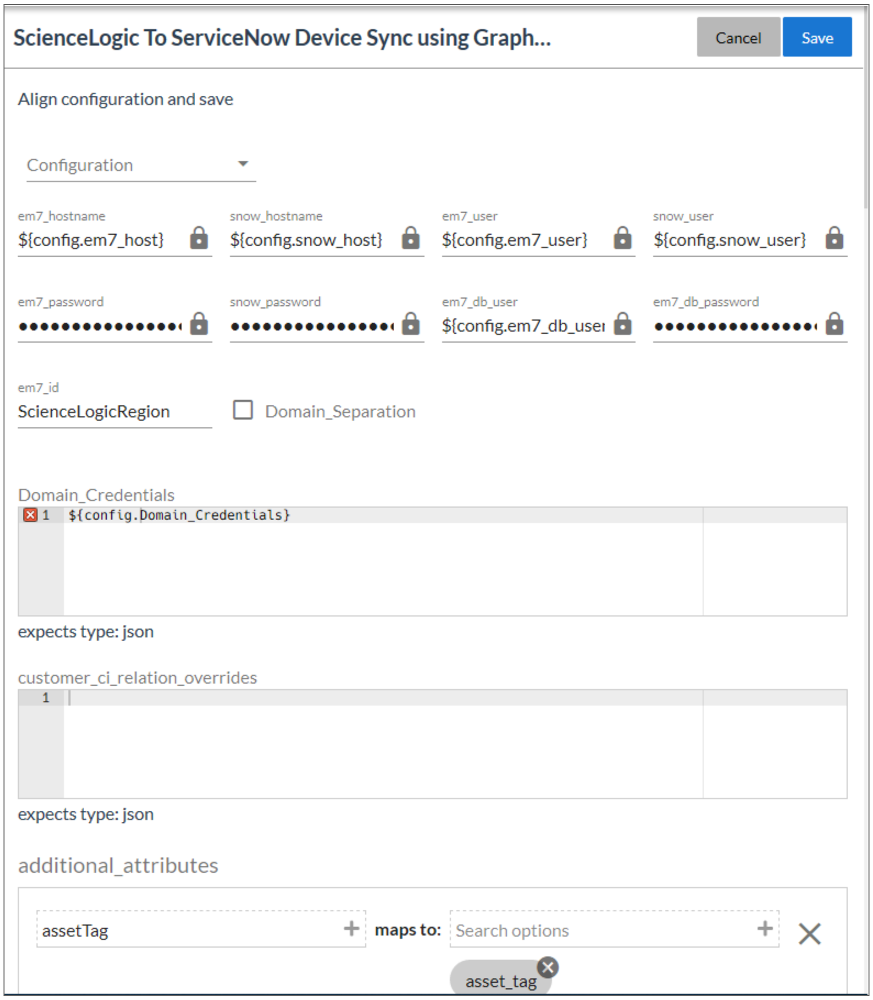
TIP: If the main pane has too many steps to see without scrolling, you can zoom in or out by clicking and holding the wheel on your mouse. You can also use the pane in the bottom right-hand corner to click on a part of the integration application that you want to see, and it will move the screen to focus on that part.

Configuring an Integration Application

Before you can run an integration application, you must align the application with a configuration from the **[Configurations]** tab. A ***configuration*** defines global variables, such as endpoints and credentials, that can be used by multiple steps and integration applications. Where relevant, you can also edit the sections for the **additional_attributes** and **mappings** parameters to update or add new mappings between SL1 and another application.

To configure an integration application:

1. On the **[Integrations]** tab, select the integration application you want to configure.
2. On the **Integration Application Editor** page, click the **[Configure]** button. The **Configuration** pane opens on the right side of the window:



ScienceLogic To ServiceNow Device Sync using Graph...

Align configuration and save

Configuration

em7_hostname: ${config.em7_host}
snow_hostname: ${config.snow_host}
em7_user: ${config.em7_user}
snow_user: ${config.snow_user}
em7_password: •••••••••••••••
snow_password: •••••••••••••••
em7_db_user: ${config.em7_db_user
em7_db_password: •••••••••••••••
em7_id: ScienceLogicRegion

☐ Domain_Separation

Domain_Credentials

```
${config.Domain_Credentials}
```

expects type: json

customer_ci_relation_overrides

```

```

expects type: json

additional_attributes

assetTag maps to: Search options

asset_tag

3. Select a configuration from the **Configuration** drop-down list to "align" to this integration application. This step is required.

4. To update the device attribute mappings for this integration application, scroll down to the section for the **additional_attributes** parameter and click the **[Add Mapping]** button to add a custom attribute, or edit an existing attribute that you want to map between SL1 and another application.

TIP: Use the **[Tab]** button to move down through the list of options in a **Mapping** dropdown list, press **[Shift]**+ **[Tab]** to move up, and press **[Enter]** to select a highlighted option.

5. To update the device class and asset mappings, scroll down to the section for the **mappings** parameter and click the **[Add Mapping]** button to add a custom class or asset, or edit an existing class or asset that you want to map between SL1 and another application.
6. As needed, edit the other configuration values for the application.

NOTE: To prevent potential issues with security and configuration, the fields related to configuration and any fields that are encrypted on the **Configuration** pane for an integration application cannot be edited.

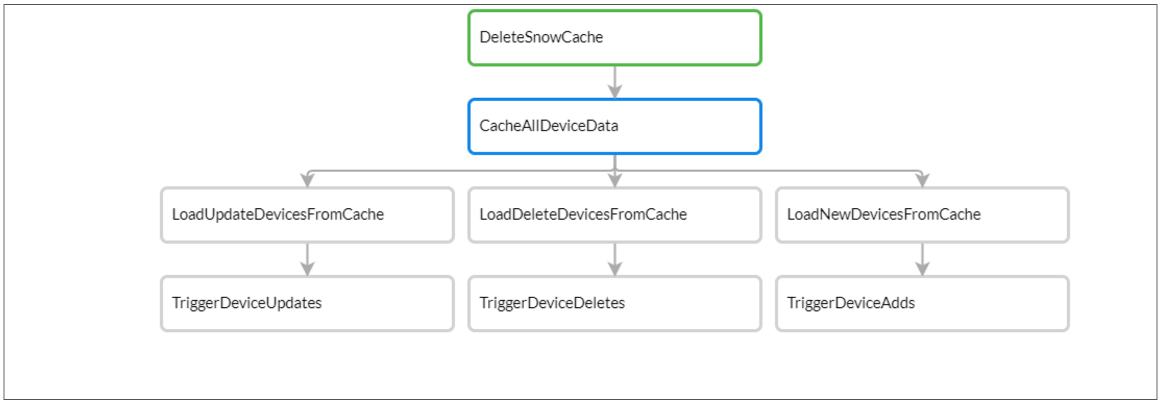
7. When you are finished, click the **[Save]** button.

NOTE: If you have created a new configuration in your own text editor, you must upload it to your Integration Service instance using the command line tool. After you upload it to the instance, it appears in the **Configuration** drop-down field on the **Configuration** pane. For more information, see the ***Integration Service for Developers*** manual.

Running or Stopping an Integration Application

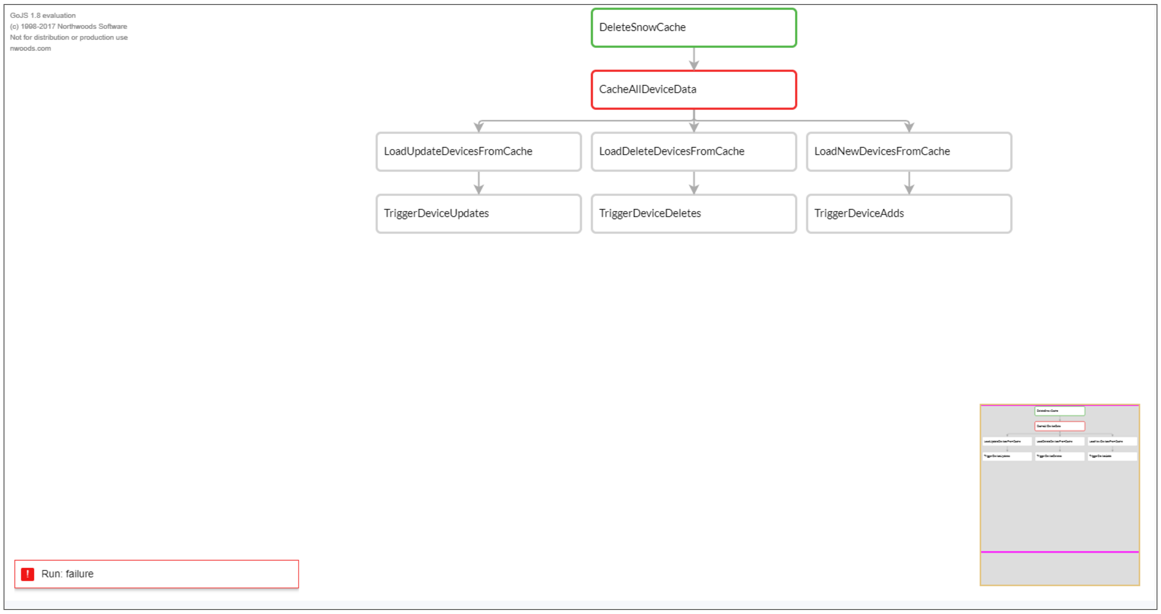
To run an integration application:

1. Click the **[Run Now]** button from the **Integrations** window or the individual application's window.
2. As the application runs, the color of the border around each step represents whether it is running, successful, or has failed:



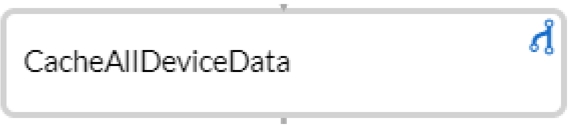
Step Color	State
Blue	Running
Green	Successful
Red	Failed

> **NOTE:** Pop-up status messages also appear in the bottom left-hand corner of the **Integration Application Editor** page to update you on the progress of the application status and alert you of any errors.



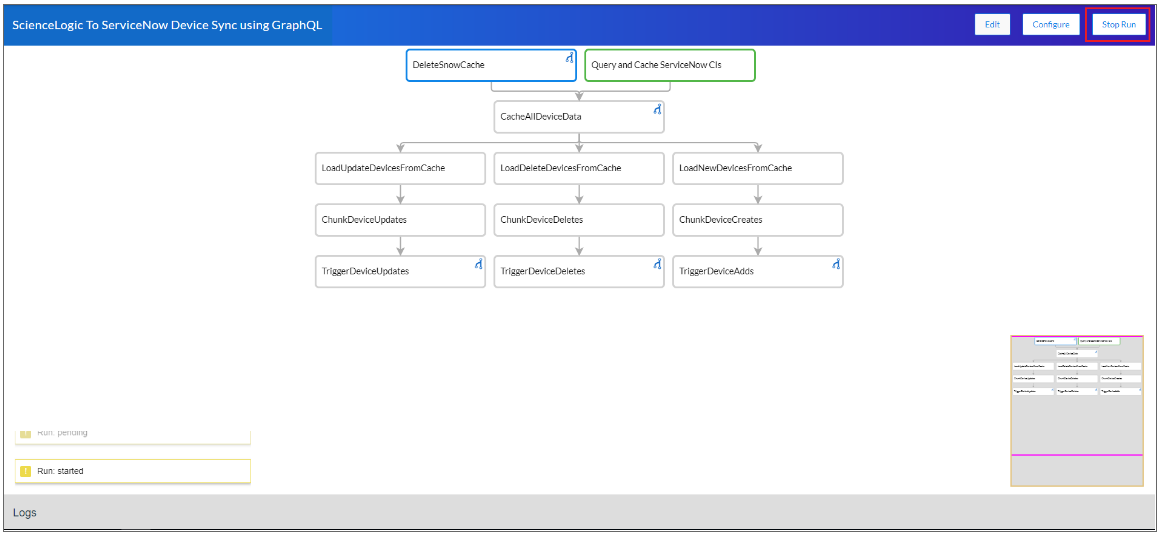
! Run: failure

3. If a step triggers a child application, a branch icon (⑂) appears in the upper right-hand corner of the step:



4. Double-click the branch icon to open the child application. Click the branch icon once to display the triggered application's run ID as a link in a pop-up window. If no run ID is present, the branch icon displays "NONE".

5. To stop the integration while it is running, click the **[Stop Run]** button. The **[Stop Run]** button is the same as the **[Run Now]** button, but toggles to **[Stop Run]** when you run the integration.



Run: pending

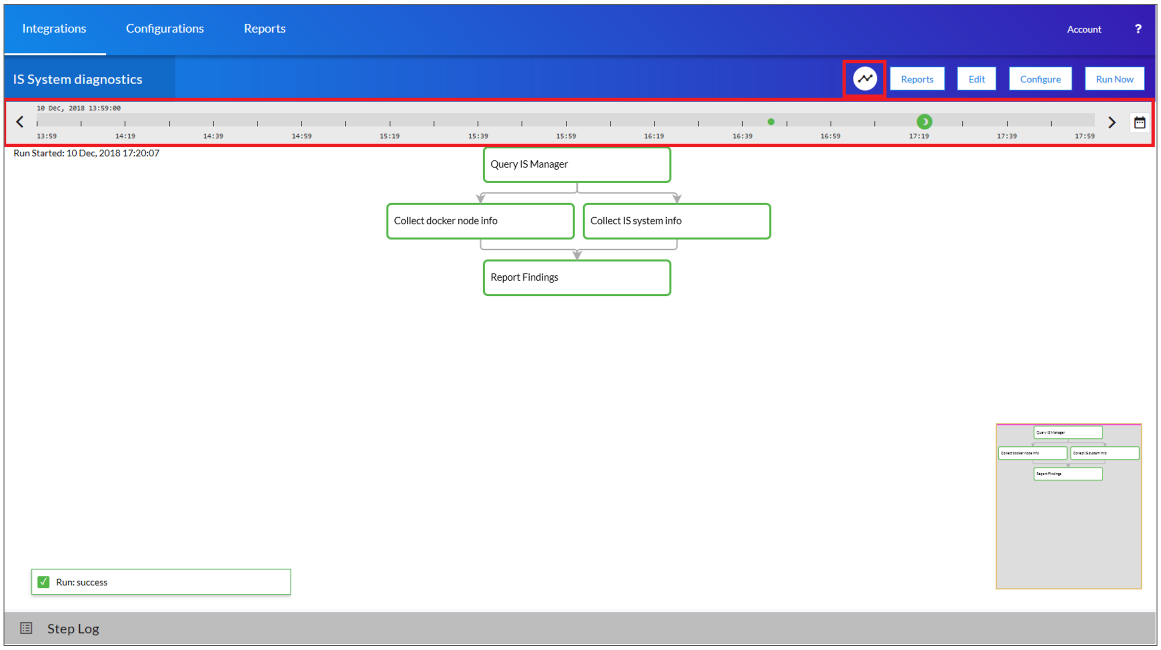
Run: started

Viewing Previous Runs of an Integration Application

When you select an integration application from the Integrations tab, the **Integration Application Editor** page appears for that integration application. The **Integration Application Editor** page contains a Timeline feature that displays a history of previous runs of that integration application.

To view and filter the Timeline:

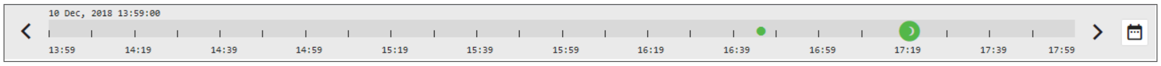
1. From an **Integration Application Editor** page, click the Timeline icon ( ). The Timeline displays above the steps for that integration application:



Run Started: 10 Dec, 2018 17:20:07

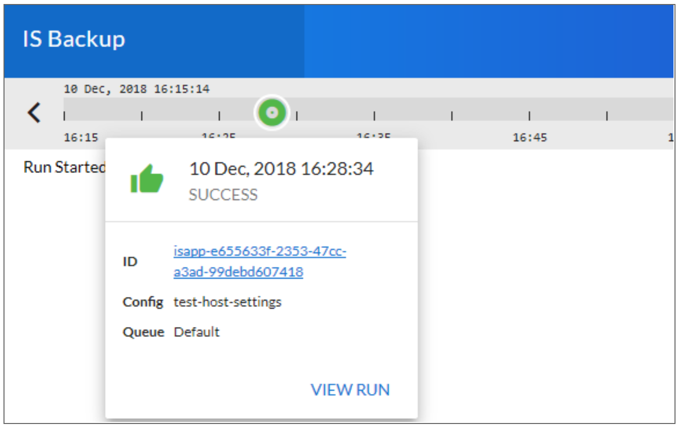
✓ Run: success

2. The default view for the Timeline shows the last two hours of runs for that integration application. Use the left arrow icon (<) and the right arrow icon (>) to move through the Timeline in 15-minute increments:



NOTE: The Timeline displays colored dots at a specific time that represent the last time this integration application was run. A green icon means a run was successful, and a red icon means that a run failed.

3. You can hover over or click an icon for a run on the Timeline to view a pop-up window that displays the run ID and the configuration and queue used for that run:

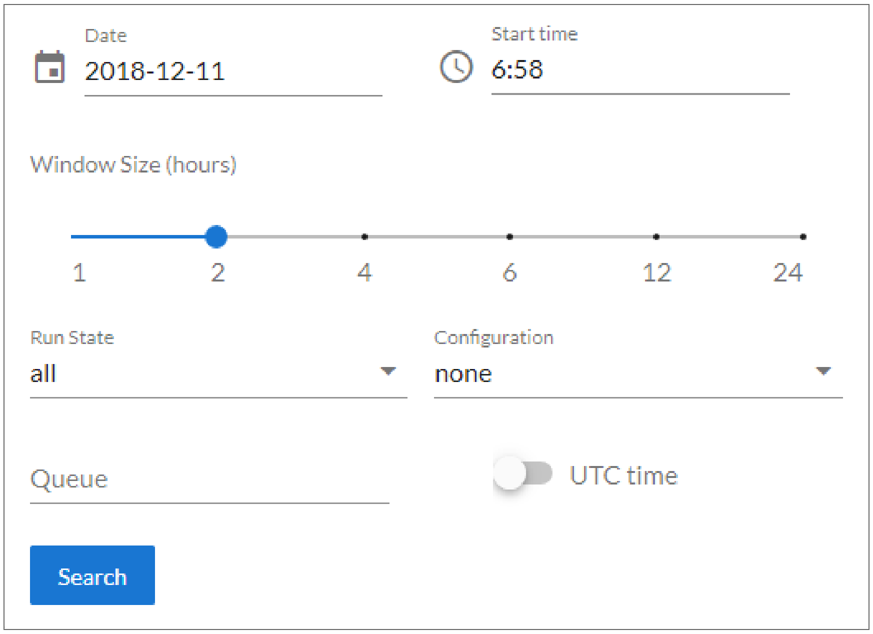


10 Dec, 2018 16:28:34
SUCCESS

ID	isapp-e655633f-2353-47cc-a3ad-99debd607418
Config	test-host-settings
Queue	Default

TIP: Click the link for the run ID or click **VIEW RUN** to open the **Integration Application Editor** page for that specific run of the integration application. On that page, you can select a step and open the **Step Log** to view any issues.

4. Click the Filter icon (📅) to filter or search the list of previous runs for this integration application. A **Filter** window appears:



Date	2018-12-11
Start time	6:58
Run State	all
Configuration	none
Queue	

5. Edit the following fields on the **Filter** window as needed:

   - ***Date***. The date for the history of previous runs you want to view. Click the field to open a pop-up calendar.
   - ***Start Time***. The starting time for the history, using local time instead of UTC time. Click the field to open a pop-up time selector.
   - ***Window Size***. The length of the history, in hours. The default history view for the Timeline is two hours.
   - ***Run State***. Select the type of previous runs you want to view. Your options include *all*, *success*, *failure*, and *pending*. The default is *all*.
   - ***Configuration***. Select a configuration file to filter for application runs using that configuration only.
   - ***Queue***. Type a queue name to filter for application runs that use that queue.
   - ***UTC Time***. Select UTC if you do not want to use local time. The Schedule feature uses UTC time.

6. Click the **[Search]** button to run the filter or search.

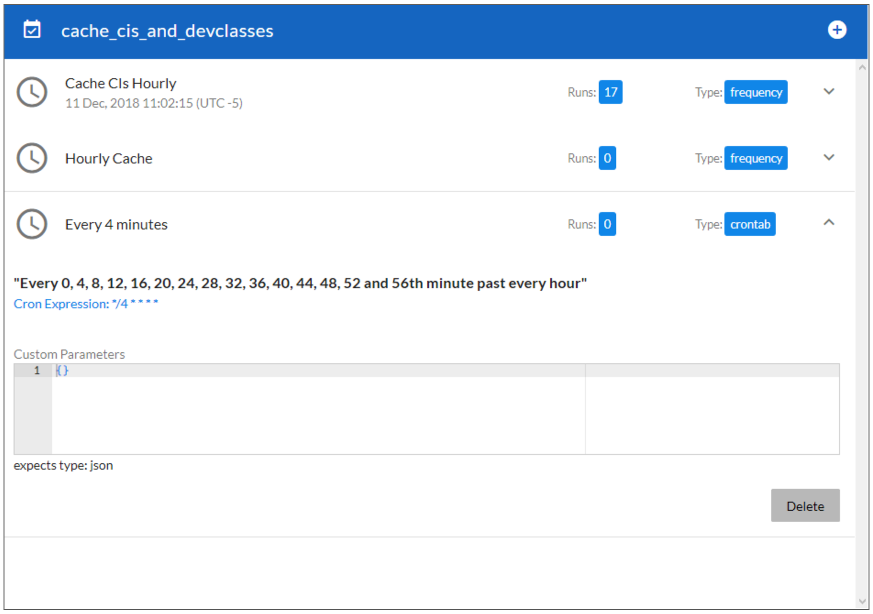
TIP: If the Timeline is open and you want to close it, click the Timeline icon ().

Scheduling an Integration Application

You can create one or more schedules for a single integration application in the Integration Service user interface. When creating the schedule, you can specify the queue and the configuration file for that integration application.

To schedule an integration application:

1. On the **[Integrations]** tab of the Integration Service user interface, click the **[Schedule]** button for the integration application you want to schedule. The initial **Schedule** window appears, displaying any existing schedules for that application:



cache_cis_and_devclasses

Cache CIs Hourly
11 Dec, 2018 11:02:15 (UTC -5)
Runs: 17 Type: frequency

Hourly Cache
Runs: 0 Type: frequency

Every 4 minutes
Runs: 0 Type: crontab

"Every 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52 and 56th minute past every hour"
Cron Expression: */4 * * * *

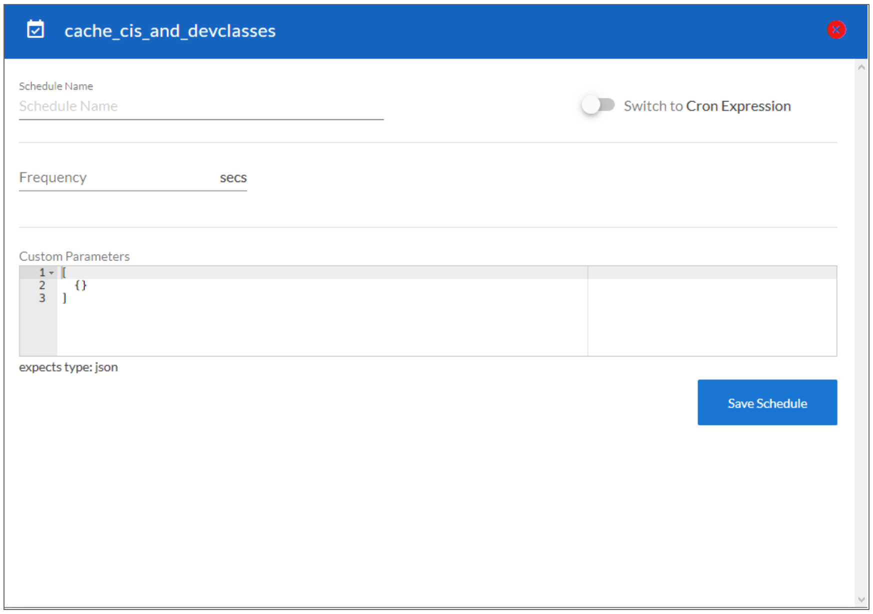
Custom Parameters

```
{}
```

expects type: json

> **NOTE:** If you set up a schedule using a cron expression, the details of that schedule display in a more readable format in this list. For example, if you set up a cron expression of */4 * * * *, the schedule on this window includes the cron expression along with an explanation of that expression: "*Every 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, and 56th minute past every hour*".

2. Click the + icon to create a schedule. A blank **Schedule** window appears:

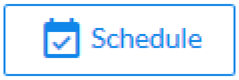


```
[
  {}
]
```

3. In the **Schedule** window, complete the following fields:

   - ***Schedule Name***. Type a name for the schedule.
   - ***Cron expression***. Select this option to schedule the integration using a cron expression. If you select this option, you can create complicated schedules based on minutes, hours, the day of the month, the month, and the day of the week. As you update the cron expression, the**Schedule** window displays the results of the expression in more readable language, such as *Expression: "Every 0 and 30th minute past every hour on the 1 and 31st of every month"*, based on ***/30 * * /30 * ***.
   - ***Frequency in seconds***.Type the number of seconds per interval that you want to run the integration.
   - ***Custom Parameters***. Type any JSON parameters you want to use for this schedule, such as information about a configuration file or mappings.

4. Click **[Save Schedule]**. The schedule is added to the list of schedules on the initial **Schedule** window. Also, on the **[Integrations]** tab, the word "Scheduled" appears in the **Scheduled** column for this integration application, and the **[Schedule]** button contains a check mark:

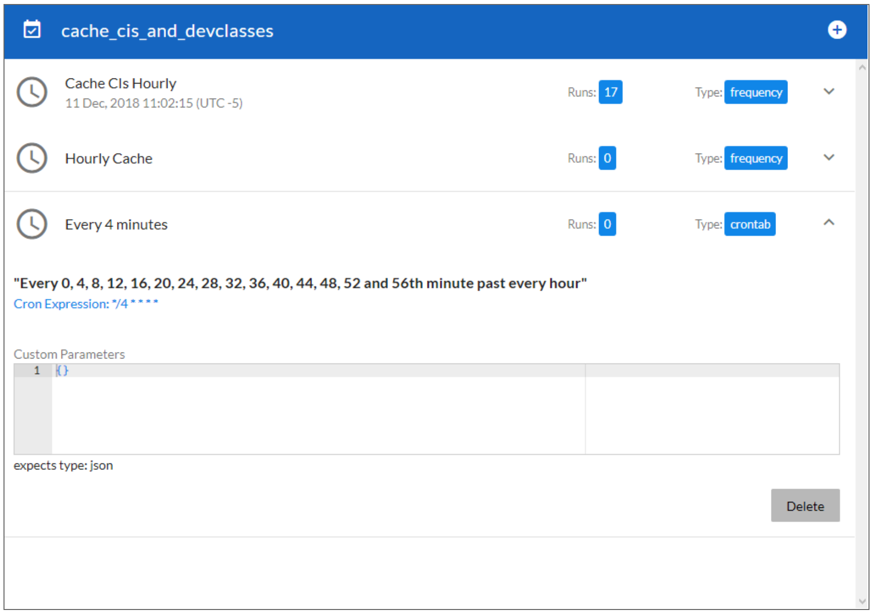


NOTE: After you create a schedule, it continues to run until you delete it. Also, you cannot edit an existing schedule, but you can delete it and create a similar schedule if needed.

To view or delete an existing schedule:

1. On the **[Integrations]** tab, click the **[Schedule]** button for the integration application that contains a schedule you want to delete. The initial **Schedule** window appears.

2. Click the down arrow icon (⌄) to view the details of an existing schedule:



cache_cis_and_devclasses

Cache CIs Hourly
11 Dec, 2018 11:02:15 (UTC -5) Runs: 17 Type: frequency

Hourly Cache Runs: 0 Type: frequency

Every 4 minutes Runs: 0 Type: crontab

"Every 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52 and 56th minute past every hour"

Cron Expression: */4 * * * *

Custom Parameters

```
{}
```

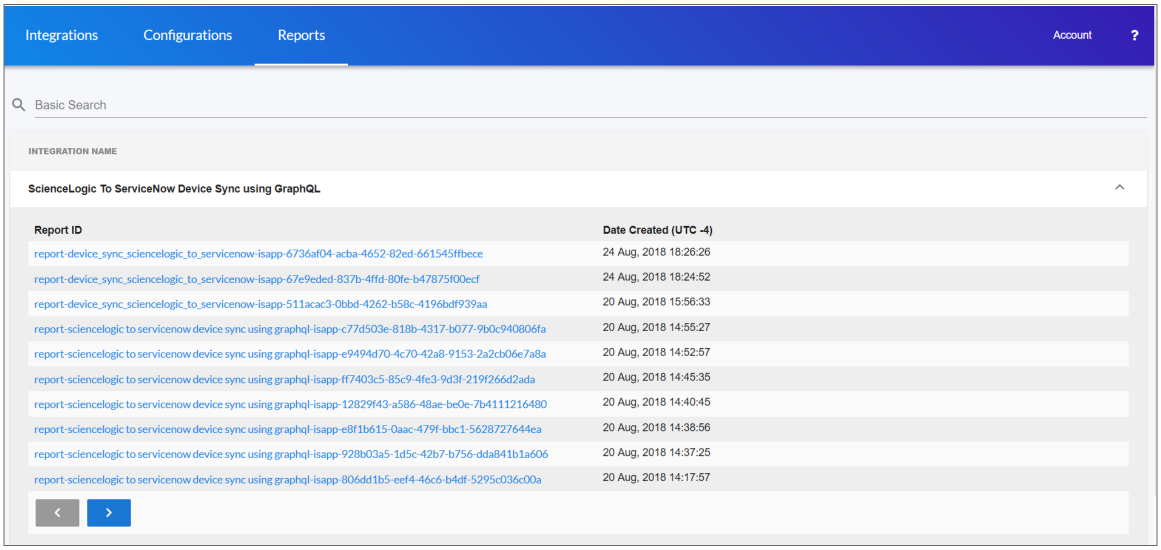
expects type: json

3. To delete the selected schedule, click the **[Delete]** button. The schedule is removed.

Viewing a Report for an Integration Application

In the Integration Service user interface, the **[Reports]** tab contains a list of reports associated with integration applications. If an integration application has the reporting feature enabled and the application supports reports, then the Integration Service will generate a report each time you run the integration application.

An individual report displays data only from the most recent run of the integration application; a report is not an aggregation of all previous runs.



ScienceLogic To ServiceNow Device Sync using GraphQL

Report ID	Date Created (UTC -4)
report-device_sync_sciencelogic_to_servicenow-isapp-6736af04-acba-4652-82ed-661545ffbece	24 Aug, 2018 18:26:26
report-device_sync_sciencelogic_to_servicenow-isapp-67e9eded-837b-4ffd-80fe-b47875f00ecf	24 Aug, 2018 18:24:52
report-device_sync_sciencelogic_to_servicenow-isapp-511acac3-0bbd-4262-b58c-4196bdf939aa	20 Aug, 2018 15:56:33
report-sciencelogic to servicenow device sync using graphql-isapp-c77d503e-818b-4317-b077-9b0c940806fa	20 Aug, 2018 14:55:27
report-sciencelogic to servicenow device sync using graphql-isapp-e9494d70-4c70-42a8-9153-2a2cb06e7a8a	20 Aug, 2018 14:52:57
report-sciencelogic to servicenow device sync using graphql-isapp-ff7403c5-85c9-4fe3-9d3f-219f266d2ada	20 Aug, 2018 14:45:35
report-sciencelogic to servicenow device sync using graphql-isapp-12829f43-a586-48ae-be0e-7b4111216480	20 Aug, 2018 14:40:45
report-sciencelogic to servicenow device sync using graphql-isapp-e8f1b615-0aac-479f-bbc1-5628727644ea	20 Aug, 2018 14:38:56
report-sciencelogic to servicenow device sync using graphql-isapp-928b03a5-1d5c-42b7-b756-dda841b1a606	20 Aug, 2018 14:37:25
report-sciencelogic to servicenow device sync using graphql-isapp-806dd1b5-eef4-46c6-b4df-5295c036c00a	20 Aug, 2018 14:17:57

NOTE: Not all integration applications generate reports. Currently, only the "ScienceLogic To ServiceNow Device Sync using GraphQL" and "IS System Diagnostics" integration applications support the generation of reports.

An integration application report includes the following fields:

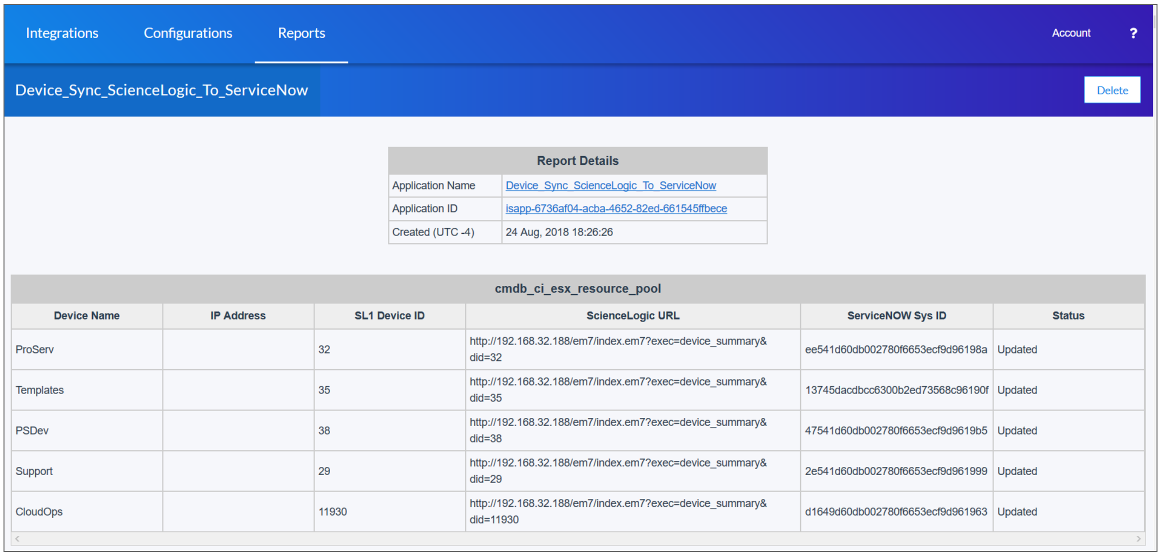
- ***Device Name***
- ***IP Address***
- ***SL1 Device ID***
- ***ScienceLogic URL***
- ***ServiceNow Sys ID***
- ***Status***. The current state of the synced item, which can include *New*, *Removed*, *Updated*, or *Unchanged*.

To view details for an integration application report:

1. On the **[Reports]** tab, click the name of the integration application to expand the list of reports for that application.

TIP: Click the arrow buttons to scroll forward and back through the list of reports.

2. Click a report name in the **Report ID** column. The **Report Details** page appears:



Report Details	
Application Name	Device_Sync_ScienceLogic_To_ServiceNow
Application ID	isapp-6736af04-acba-4652-82ed-661545ffbece
Created (UTC -4)	24 Aug, 2018 18:26:26

cmdb_ci_esx_resource_pool

Device Name	IP Address	SL1 Device ID	ScienceLogic URL	ServiceNOW Sys ID	Status
ProServ		32	http://192.168.32.188/em7/index.em7?exec=device_summary&did=32	ee541d60db002780f6653ecf9d96198a	Updated
Templates		35	http://192.168.32.188/em7/index.em7?exec=device_summary&did=35	13745dacdbcc6300b2ed73568c96190f	Updated
PSDev		38	http://192.168.32.188/em7/index.em7?exec=device_summary&did=38	47541d60db002780f6653ecf9d9619b5	Updated
Support		29	http://192.168.32.188/em7/index.em7?exec=device_summary&did=29	2e541d60db002780f6653ecf9d961999	Updated
CloudOps		11930	http://192.168.32.188/em7/index.em7?exec=device_summary&did=11930	d1649d60db002780f6653ecf9d961963	Updated

3. To view the detail page for the integration application on the **[Integrations]** tab, click the **Application Name** link.

TIP: From the detail page for the integration application, click the **[Reports]** button to return to the **[Reports]** tab.

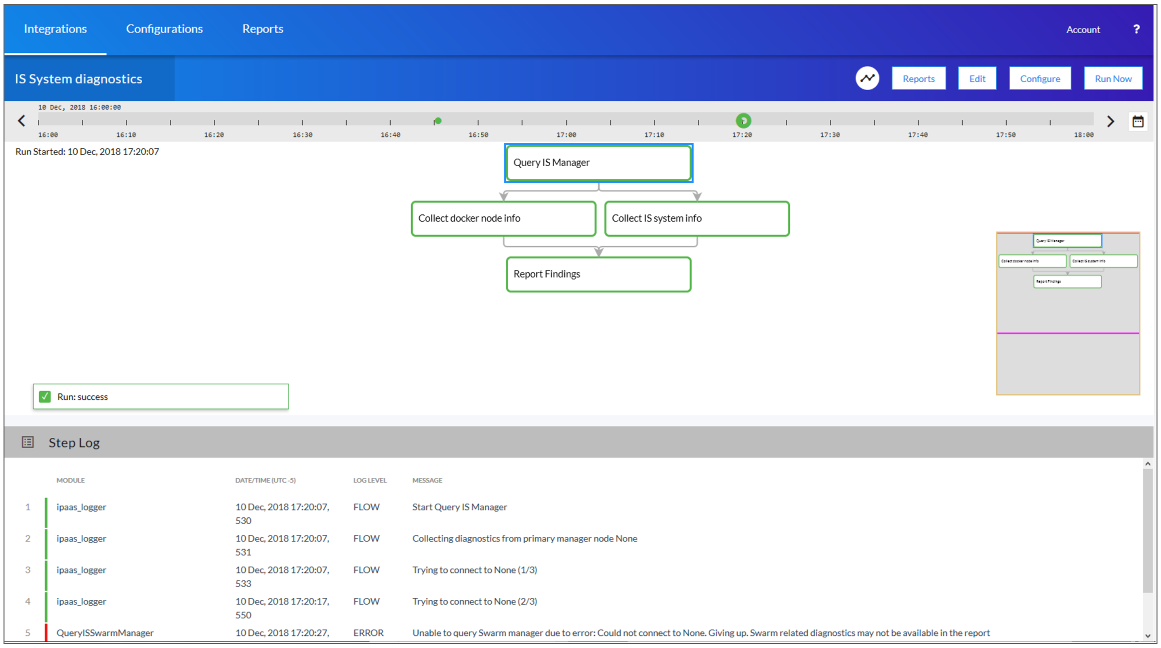
4. To view the specific run-time instance for the integration application that generated the report, click the **Application ID** link.
5. To delete a report, click the **[Delete]** button. Click **[OK]** to delete the report.

Viewing Logs for an Integration Application

You can view logs for each step in an integration application on the **Integration Application Editor** page in the Integration Service user interface.

To view logs for the steps of an integration application:

1. From the **[Integrations]** tab, select an integration application. The **Integration Application Editor** page appears.
2. Select a step in the integration application. Required.
3. Click to open the **Step Log** in the bottom left-hand corner of the screen. The **Step Log** pane appears at the bottom of the page, and it displays status messages for the selected step:



Run Started: 10 Dec, 2018 17:20:07

☑ Run: success

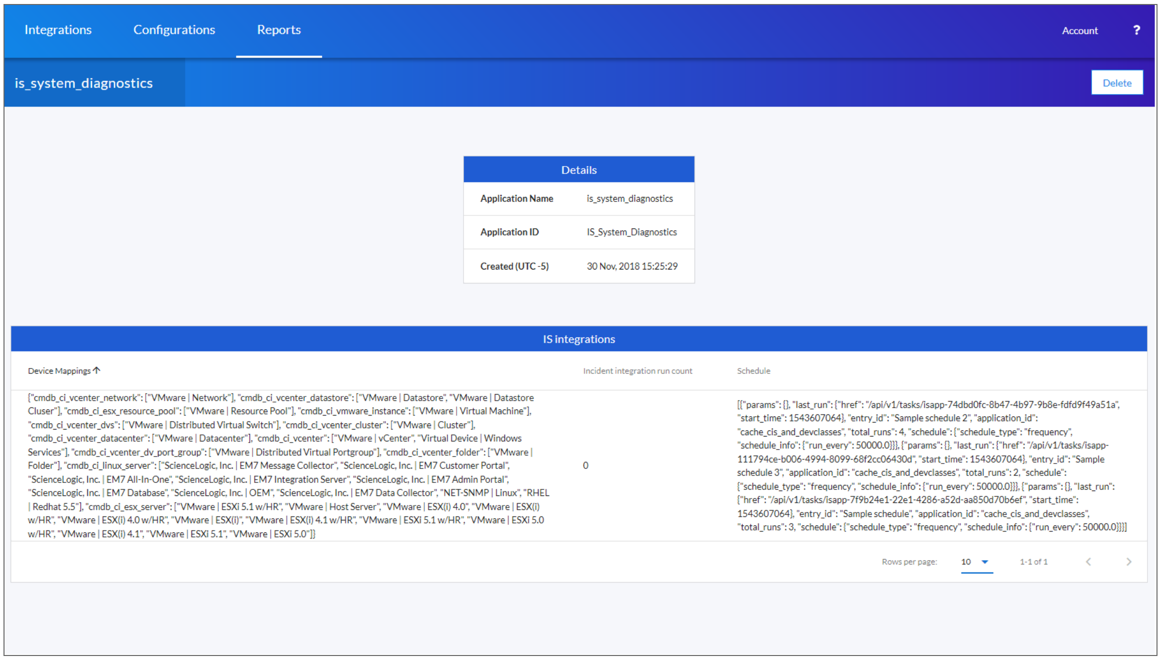
	MODULE	DATE/TIME (UTC -5)	LOG LEVEL	MESSAGE
1	ipaas_logger	10 Dec, 2018 17:20:07, 530	FLOW	Start Query IS Manager
2	ipaas_logger	10 Dec, 2018 17:20:07, 531	FLOW	Collecting diagnostics from primary manager node None
3	ipaas_logger	10 Dec, 2018 17:20:07, 533	FLOW	Trying to connect to None (1/3)
4	ipaas_logger	10 Dec, 2018 17:20:17, 550	FLOW	Trying to connect to None (2/3)
5	QueryISSwarmManager	10 Dec, 2018 17:20:27,	ERROR	Unable to query Swarm manager due to error: Could not connect to None. Giving up. Swarm related diagnostics may not be available in the report

TIP: Click the gray area of the **Logs** pane to close the pane.

Viewing System Diagnostics

The "IS System Diagnostics" integration application lets you view platform diagnostics for the Integration Service. You can use the information displayed in these diagnostics to help you troubleshoot issues with the different tools used by the Integration Service.

Running the "IS System Diagnostics" integration application in the Integration Service user interface generates a report that you can access on the **[Reports]** tab:



Details	
Application Name	is_system_diagnostics
Application ID	IS_System_Diagnostics
Created (UTC -5)	30 Nov, 2018 15:25:29

<table>
<tr><th>Device Mappings ↑</th><th>Incident integration run count</th><th>Schedule</th></tr>
<tr><td>{"cmdb_ci_vcenter_network": ["VMware | Network"], "cmdb_ci_vcenter_datastore": ["VMware | Datastore", "VMware | Datastore Cluser"], "cmdb_ci_esx_resource_pool": ["VMware | Resource Pool"], "cmdb_ci_vmware_instance": ["VMware | Virtual Machine"], "cmdb_ci_vcenter_dvs": ["VMware | Distributed Virtual Switch"], "cmdb_ci_vcenter_cluster": ["VMware | Cluster"], "cmdb_ci_vcenter_datacenter": ["VMware | Datacenter"], "cmdb_ci_vcenter": ["VMware | vCenter", "Virtual Device | Windows Services"], "cmdb_ci_vcenter_dv_port_group": ["VMware | Distributed Virtual Portgroup"], "cmdb_ci_vcenter_folder": ["VMware | Folder"], "cmdb_ci_linux_server": ["ScienceLogic, Inc. | EM7 Message Collector", "ScienceLogic, Inc. | EM7 Customer Portal", "ScienceLogic, Inc. | EM7 All-In-One", "ScienceLogic, Inc. | EM7 Integration Server", "ScienceLogic, Inc. | EM7 Admin Portal", "ScienceLogic, Inc. | EM7 Database", "ScienceLogic, Inc. | OEM", "ScienceLogic, Inc. | EM7 Data Collector", "NET-SNMP | Linux", "RHEL | Redhat 5.5"], "cmdb_ci_esx_server": ["VMware | ESXi 5.1 w/HR", "VMware | Host Server", "VMware | ESX(i) 4.0", "VMware | ESX(i) w/HR", "VMware | ESX(i) 4.0 w/HR", "VMware | ESX(i)", "VMware | ESX(i) 4.1 w/HR", "VMware | ESXi 5.1 w/HR", "VMware | ESXi 5.0 w/HR", "VMware | ESX(i) 4.1", "VMware | ESXi 5.1", "VMware | ESXi 5.0"]}</td><td>0</td><td>[{"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-74dbd0fc-8b47-4b97-9b8e-fdfd9f49a51a", "start_time": 1543607064}, "entry_id": "Sample schedule 2", "application_id": "cache_cis_and_devclasses", "total_runs": 4, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}, {"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-111794ce-b006-4994-8099-68f2cc06430d", "start_time": 1543607064}, "entry_id": "Sample schedule 3", "application_id": "cache_cis_and_devclasses", "total_runs": 2, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}, {"params": {}, "last_run": {"href": "/api/v1/tasks/isapp-7f9b24e1-22e1-4286-a52d-aa850d70b6ef", "start_time": 1543607064}, "entry_id": "Sample schedule", "application_id": "cache_cis_and_devclasses", "total_runs": 3, "schedule": {"schedule_type": "frequency", "schedule_info": {"run_every": 50000.0}}}]</td></tr>
</table>

This diagnostic report displays overall Integration Service settings, such as the Integration Service version, Docker version, kernel version, hostname, cluster settings, scheduled applications, CPU and memory statistics, installation date, and cache information.

TIP: If you are using a specific integration application that you want to monitor with the "IS System Diagnostics" integration application, click the **[Configure]** button and type the name of that integration application in the ***incident_create_app*** field of the **Configuration** pane.

Backing up Data for Disaster Recovery

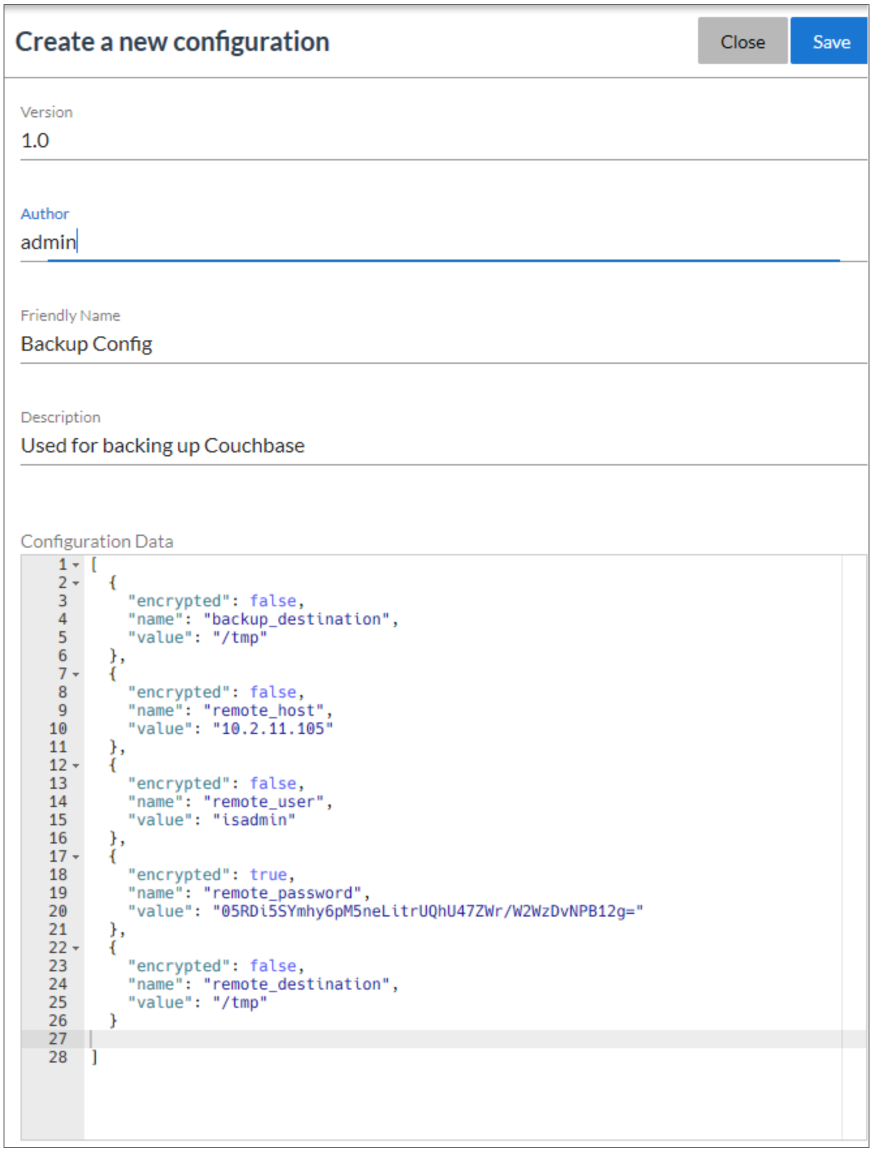
You can use a Disaster Recovery option that enables the Integration Service to back up and recover data in the Couchbase database.

This option uses the "IS Backup" integration application in the Integration Service user interface to create a backup file and send that file using secure copy protocol (SCP) to a destination system. You can then use the "IS Restore" integration application get a backup file from the remote system and restore its content.

Creating a Backup

To create a backup:

1. To add the relevant configuration information, create a new configuration or update an existing configuration on the **[Configurations]** tab of the Integration Service user interface:



Create a new configuration

Version
1.0

Author
admin

Friendly Name
Backup Config

Description
Used for backing up Couchbase

Configuration Data

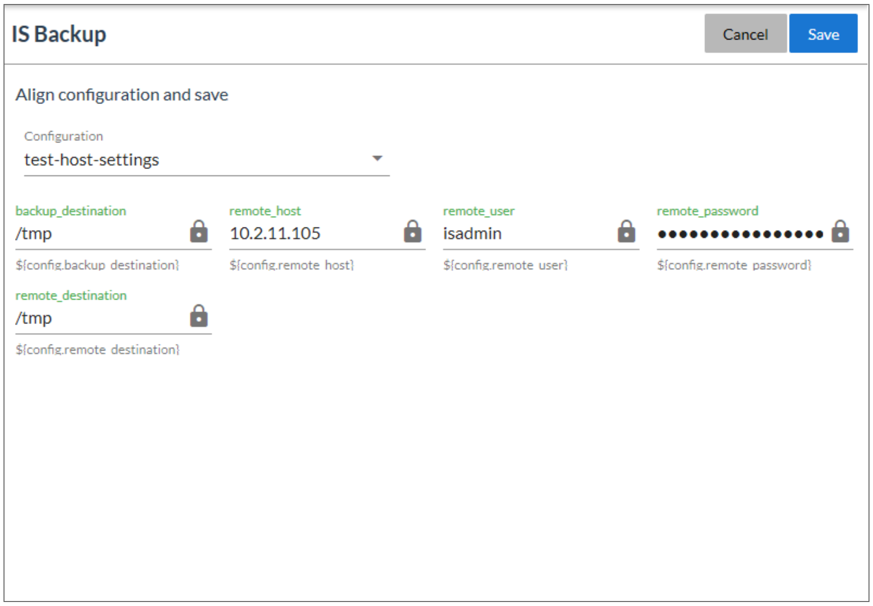
```
[
  {
    "encrypted": false,
    "name": "backup_destination",
    "value": "/tmp"
  },
  {
    "encrypted": false,
    "name": "remote_host",
    "value": "10.2.11.105"
  },
  {
    "encrypted": false,
    "name": "remote_user",
    "value": "isadmin"
  },
  {
    "encrypted": true,
    "name": "remote_password",
    "value": "05RDi5SYmhy6pM5neLitrUQhU47ZWr/W2WzDvNPB12g="
  }, 
  {
    "encrypted": false,
    "name": "remote_destination",
    "value": "/tmp"
  }

]
```

2. In the configuration file, provide values for the following fields:

    - ***backup_destination***. The location where the backup file is created.
    - ***remote_host***. The hostname for the remote location where you want to send the backup file via SCP from the *backup_destination* location.
    - ***remote_user***. The user login for the remote location.
    - ***remote_password***. The user password for the remote location. Encrypt this value.
    - ***remote_destination***. The remote location where the integration application will send the backup file.

3. Click the **[Save]** button to save the new or updated configuration file.
4. Go to the **[Integrations]** tab and select the "IS Backup" integration application. The **IS Backup Integration Editor** page appears.
5. To change the configuration file used by this integration application, click the **[Configure]** button. The **Configuration** pane appears:



Align configuration and save

Configuration: test-host-settings

backup_destination: /tmp
${config.backup_destination}

remote_host: 10.2.11.105
${config.remote_host}

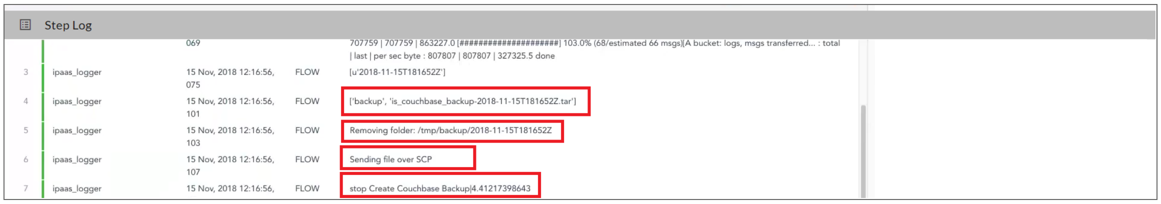
remote_user: isadmin
${config.remote_user}

remote_password: ●●●●●●●●●●●●●●●●
${config.remote_password}

remote_destination: /tmp
${config.remote_destination}

6. Select a configuration from the ***Configuration*** drop-down and click the **[Save]** button.
7. On the **IS Backup Integration Editor** page, click the **[Run Now]** button. When the application completes, a file named "is_couchbase_backup-<*date*>.tar" is added to the remote server in the specified remote backup destination.

8. To ensure that the backup was created, click to open the ***Step Log*** section and look for entries related to backup. Make a note of the of the backup file name, which you will need when you run the "IS Restore" integration application:



Step Log

```
                     069                          707759 | 707759 | 863227.0 [####################] 103.0% (68/estimated 66 msgs)[A bucket: logs, msgs transferred... : total
                                                  | last | per sec byte : 807807 | 807807 | 327325.5 done
3   ipaas_logger     15 Nov, 2018 12:16:56,  FLOW  [u'2018-11-15T181652Z']
                     075
4   ipaas_logger     15 Nov, 2018 12:16:56,  FLOW  ['backup', 'is_couchbase_backup-2018-11-15T181652Z.tar']
                     101
5   ipaas_logger     15 Nov, 2018 12:16:56,  FLOW  Removing folder: /tmp/backup/2018-11-15T181652Z
                     103
6   ipaas_logger     15 Nov, 2018 12:16:56,  FLOW  Sending file over SCP
                     107
7   ipaas_logger     15 Nov, 2018 12:16:56,  FLOW  stop Create Couchbase Backup|4.41217398643
```

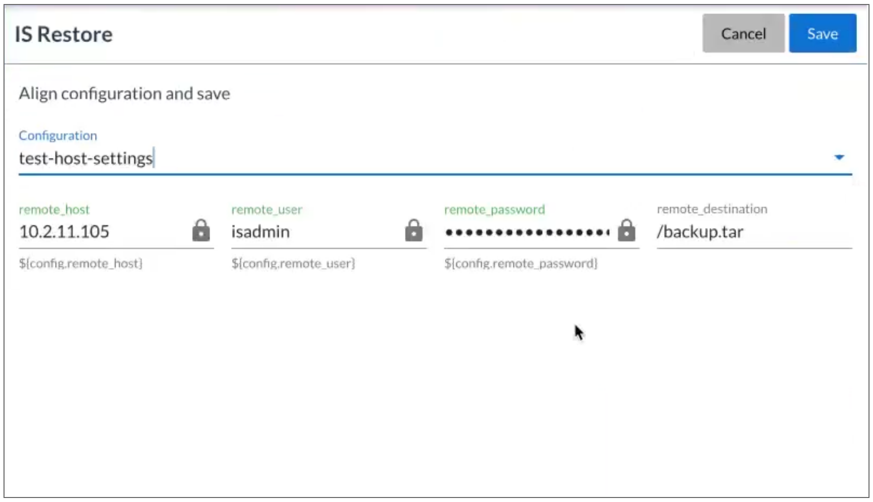
TIP: You can schedule the "IS Backup" integration application to run on a regular basis, or you can run the application as needed. To schedule the application, click the **[Schedule]** button for the "IS Backup" application on the **[Integrations]** tab. For more information, see *Scheduling an Integration Application*.

Restoring a Backup

After you have created a backup using the "IS Backup" integration application in the Integration Service user interface, you can use the "IS Restore" integration application to restore that file.

To restore a backup:

1. In the Integration Service user interface, go to the **[Interfaces]** tab and select the **IS Restore** integration application. The **IS Restore** page appears.
2. To update the configuration file used by this integration application, click the **[Configure]** button. The **Configuration** pane appears:



IS Restore

Align configuration and save

Configuration: test-host-settings

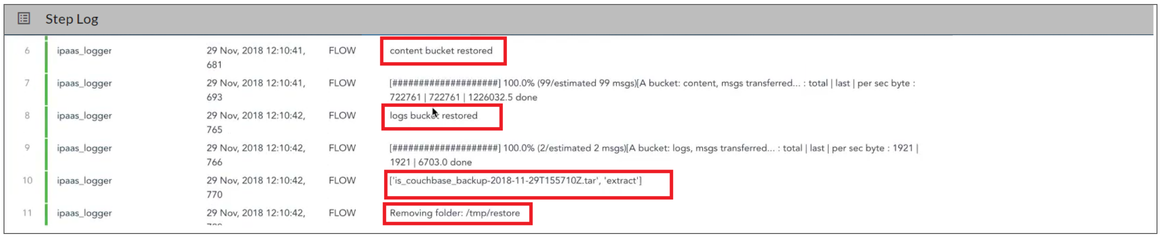
remote_host: 10.2.11.105
${config.remote_host}

remote_user: isadmin
${config.remote_user}

remote_password: ••••••••••••••••••
${config.remote_password}

remote_destination: /backup.tar

3. Select a configuration from the ***Configuration*** drop-down.
4. In the ***remote_destination*** field, type the name of the backup file created by the "IS Backup" integration application.
5. Click the **[Save]** button.
6. On the **IS Restore Integration Editor** page, click the **[Run Now]** button.
7. To ensure that the backup was created, click to open the ***Step Log*** section and look for entries related to restoring the backup:



Step Log

```
6   ipaas_logger   29 Nov, 2018 12:10:41,   FLOW   content bucket restored
                   681
7   ipaas_logger   29 Nov, 2018 12:10:41,   FLOW   [####################] 100.0% (99/estimated 99 msgs)[A bucket: content, msgs transferred... : total | last | per sec byte :
                   693                             722761 | 722761 | 1226032.5 done
8   ipaas_logger   29 Nov, 2018 12:10:42,   FLOW   logs bucket restored
                   765
9   ipaas_logger   29 Nov, 2018 12:10:42,   FLOW   [####################] 100.0% (2/estimated 2 msgs)[A bucket: logs, msgs transferred... : total | last | per sec byte : 1921 |
                   766                             1921 | 6703.0 done
10  ipaas_logger   29 Nov, 2018 12:10:42,   FLOW   ['is_couchbase_backup-2018-11-29T155710Z.tar', 'extract']
                   770
11  ipaas_logger   29 Nov, 2018 12:10:42,   FLOW   Removing folder: /tmp/restore
```

Chapter

5

Managing Configurations

Overview

On the **[Configurations]** tab of the Integration Service user interface, you can create a configuration to define global variables that all steps and integration applications can use.

This chapter covers the following topics:

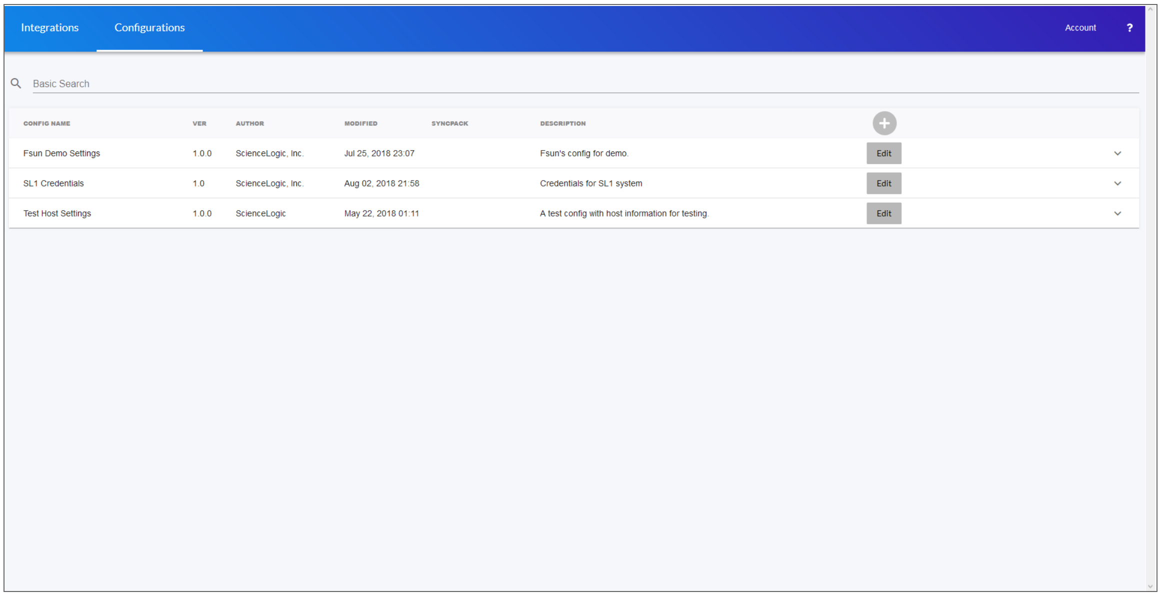
What is a Configuration?

A ***configuration*** is a stand-alone JSON file that lives on the Integration Service system. A configuration defines global variables that can be used by all steps and integration applications.

After you create the configuration, it appears in the **Configuration** drop-down field on the **Configuration** pane of the **[Integrations]** tab. Before you can run an integration application, you must select a configuration and "align" that configuration with the integration application.

You can include the **config.** prefix with a variable to tell the Integration Service to use a configuration file to resolve the variable. If you want to re-use the same settings between applications, such as hostname and credentials, define configuration variables.

The **Configuration Registry** page displays a list of available configurations. From this page you can create and edit configurations:



CONFIG NAME	VER	AUTHOR	MODIFIED	SYNCPACK	DESCRIPTION
Fsun Demo Settings	1.0.0	ScienceLogic, Inc.	Jul 25, 2018 23:07		Fsun's config for demo.
SL1 Credentials	1.0	ScienceLogic, Inc.	Aug 02, 2018 21:58		Credentials for SL1 system
Test Host Settings	1.0.0	ScienceLogic	May 22, 2018 01:11		A test config with host information for testing.

For each configuration, the page displays:

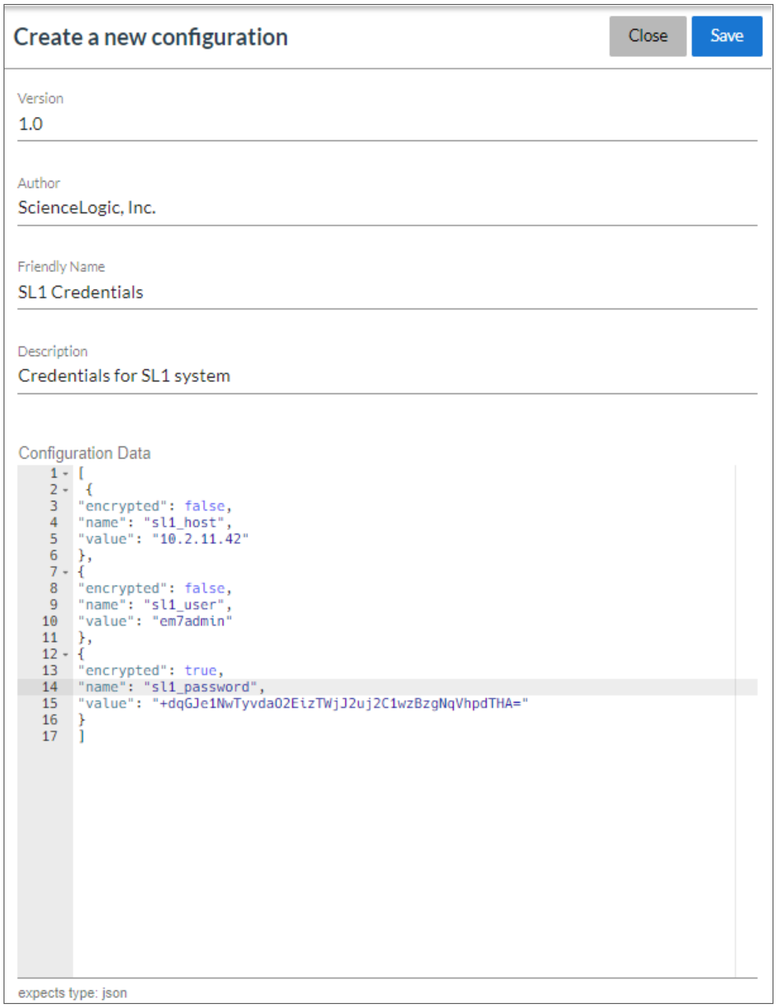
- ***Config Name***. Name of the configuration.
- ***Ver***. Version of the configuration.
- ***Author***. User or organization that created the configuration.
- ***Modified***. The date and time the configuration was created or last edited.
- ***SyncPack***. The SyncPack associated with the configuration.
- ***Description***. A brief description of the configuration.

NOTE: The Integration Service includes an example configuration on the **[Configurations]** tab called "Test Host Settings" that you can use as a template.

Creating a Configuration

To create a new configuration:

1. Navigate to the **Configuration Registry** page (the **[Configurations]** tab).
2. Click the plus icon (⊕). The **Create a new configuration** pane appears:



Create a new configuration

Version
1.0

Author
ScienceLogic, Inc.

Friendly Name
SL1 Credentials

Description
Credentials for SL1 system

Configuration Data

```
[
 {
"encrypted": false,
"name": "sl1_host",
"value": "10.2.11.42"
},
{
"encrypted": false,
"name": "sl1_user",
"value": "em7admin"
},
{
"encrypted": true,
"name": "sl1_password",
"value": "+dqGJe1NwTyvdaO2EizTWjJ2uj2C1wzBzgNqVhpdTHA="
}
]
```

expects type: json

3. Complete the following fields:

   - ***Version***. Version of the configuration.
   - ***Author***. User or organization that created the configuration.
   - ***Friendly Name***. Name of the configuration.
   - ***Description***. A brief description of the configuration.

4. In the ***Configuration Data*** field, specify the variable definitions:

   For example, you could add the following JSON code to the ***Configuration Data*** field:

   ```
   [
     {
       "encrypted": false,
       "name": "em7_host",
       "value": "10.2.11.42"
     },
     {
       "encrypted": false,
       "name": "em7_user",
       "value": "em7admin"
     },
     {
       "encrypted": true,
       "name": "em7_password",
       "value": "+dqGJe1NwTyvdaO2EizTWjJ2uj2C1wzBzgNqVhpdTHA="
     }
   ]
   ```

5. When creating a configuration variable, note the syntax:

   - The configuration file is surrounded by square brackets.
   - Each variable definition is surrounded by curly braces.
   - Each key name is surrounded by double-quotes and followed by a colon, while each value is surrounded by double-quotes and followed by a comma.
   - Each key:value pair in the definition is separated with a comma after the closing curly brace. The last key:value pair should not include a comma.

6. To create a configuration variable, define the following keys:

   - ***encrypted***. Specifies whether the value will appear in plain text or encrypted in this JSON file. If you set this to "true", when the value is uploaded, the Integration Service encrypts the value of the variable. The plain text value cannot be retrieved again by an end user. The encryption key is unique to each Integration Service system. The value is followed by a comma.
   - ***name***. Specifies the name of the configuration file, without the JSON suffix. This value appears in the user interface. The value is surrounded by double-quotes and followed by a comma.
   - ***value***. Specifies the value to assign to the variable. The value is surrounded by double-quotes and followed by a comma.

7. Repeat steps 5 and 6 for each configuration variable.

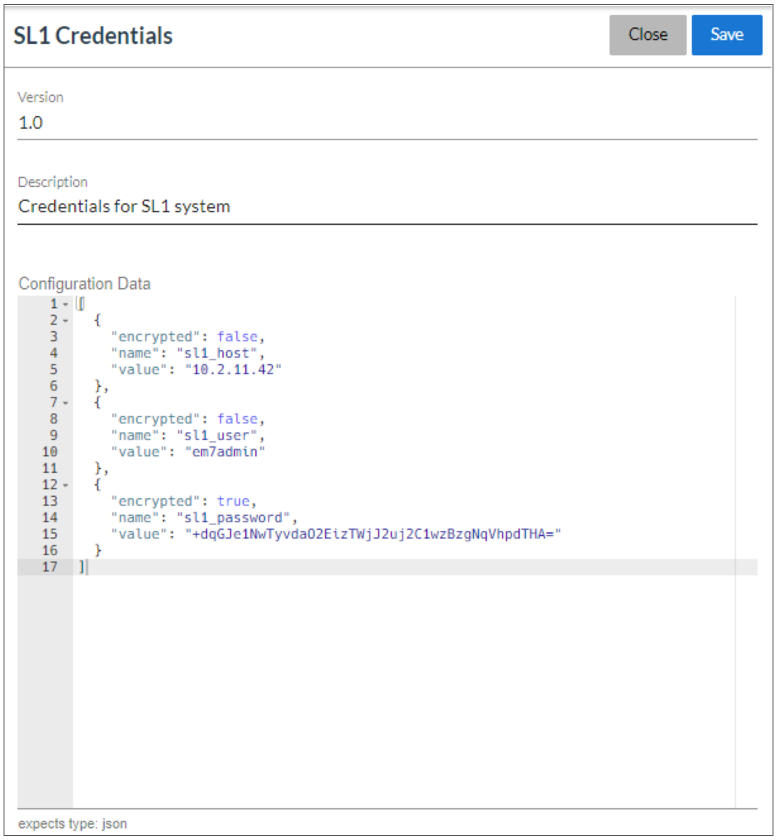
8. Click the **[Save]**button.

NOTE: In a step, you can include the **config.** prefix with a variable to tell the Integration Service system to look in a configuration file to resolve the variable.

Editing a Configuration

To edit an existing configuration:

1. Navigate to the **Configuration Registry** page (the **[Configurations]** tab).
2. Click the **[Edit]**button for the configuration you want to edit. The **Configuration Editor** pane appears:



SL1 Credentials

Version

1.0

Description

Credentials for SL1 system

Configuration Data

```
[
  {
    "encrypted": false,
    "name": "sl1_host",
    "value": "10.2.11.42"
  },
  {
    "encrypted": false,
    "name": "sl1_user",
    "value": "em7admin"
  },
  {
    "encrypted": true,
    "name": "sl1_password",
    "value": "+dqGJe1NwTyvda02EizTWjJ2uj2C1wzBzgNqVhpdTHA="
  }
]
```

expects type: json

3. On the **Configuration Editor** pane, edit the values in the following fields as needed:

   - ***Version***. Version of the configuration.
   - ***Description***. A brief description of the configuration.
   - ***Configuration Data***. Definition of each global variable. You can edit an existing definition and add definitions.

4. Click **[Save]** to save your changes.

Chapter 6

Viewing Logs in the Integration Service

Overview

This chapter describes the different types of logging available in the Integration Service.

This chapter covers the following topics:

Logging Data in the Integration Service

The Integration Service allows you to view log data locally, remotely, or through Docker.

Local Logging

The Integration Service writes logs to files on a host system directory. Each of the main components, such as the process manager or Celery workers, and each application that is run on the platform, generates a log file. The application log files use the application name for easy consumption and location.

In a clustered environment, the logs must be written to the same volume or disk being used for persistent storage. This ensures that all logs are gathered from all hosts in the cluster onto a single disk, and that each application log can contain information from separately located, disparate workers.

You can also implement log features such as rolling, standard out, level, and location setting, and you can configure these features with their corresponding environment variable or setting in a configuration file.

Remote Logging

If you use your own existing logging server, such as Syslog, Splunk, or Logstash, the Integration Service can route its logs to a customer-specified location. To do so, attach your service, such as logspout, to the Microservice stack and configure your service to route all logs to the server of your choice.

CAUTION: Although the Integration Service supports logging to these remote systems, ScienceLogic does not officially own or support the configuration of the remote logging locations. Use the logging to a remote system feature at your own discretion.

Viewing Logs in Docker

You can use the Docker command line to view the logs of any current running service in the Integration Service cluster. To view the logs of any service, run the following command:

```
docker service logs -f iservices_service_name
```

Some common examples include the following:

```
docker service logs -f iservices_couchbase

docker service logs -f iservices_steprunner

docker service logs -f iservices_contentapi
```

Logging Configuration

The following table describes the variables and configuration settings related to logging in the Integration Service:

Environment Variable/Config Setting	Description	Default Setting
logdir	The director to which logs will be written.	/var/log/iservices
stdoutlog	Whether logs should be written to standard output (stdout).	True
loglevel	Log level setting for Integration Service application modules.	debug/info (varies between development and product environments)
celery_log_level	The log level for Celery-related components and modules.	debug/info (varies between development and product environments)
log_rollover_size	Size of the Integration Service logs to keep before rolling over.	10 MB
log_rollover_max_files	Max number of log files to keep when rolling over.	5

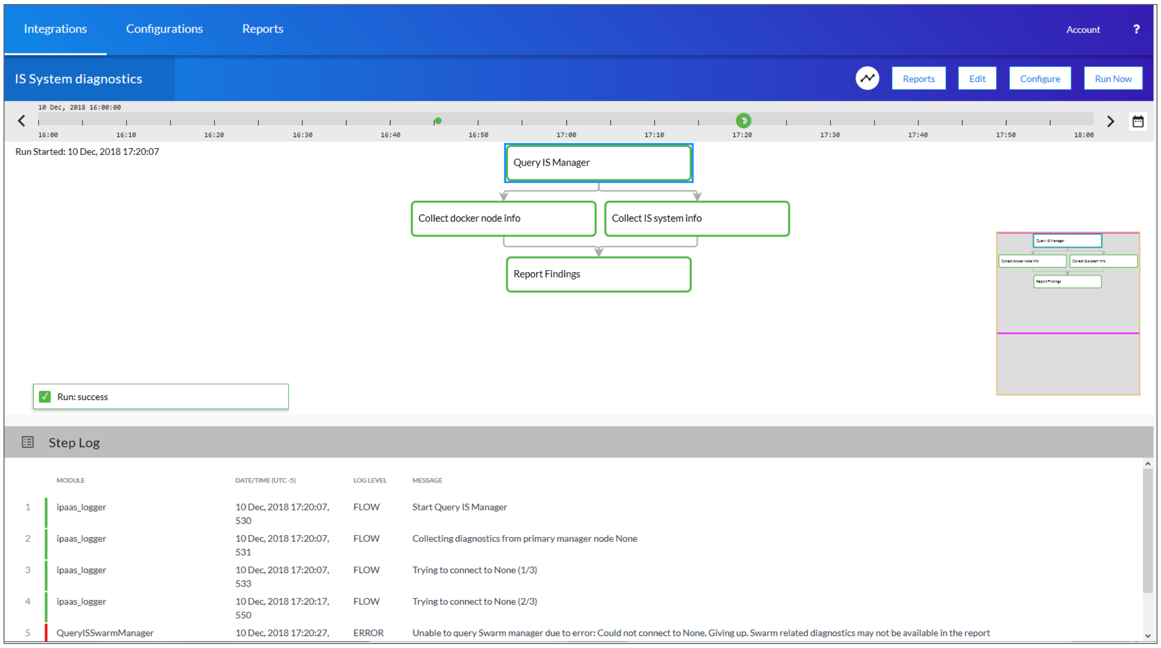
Viewing Logs in the User Interface

You can view logs for each step in an integration application on the **Integration Application Editor** page in the Integration Service user interface.

To view logs for the steps of an integration application:

1. From the **[Integrations]** tab, select an integration application. The **Integration Application Editor** page appears.
2. Select a step in the integration application. Required.

3. Click to open the **Step Log** in the bottom left-hand corner of the screen. The **Step Log** pane appears at the bottom of the page, and it displays status messages for the selected step:



Run Started: 10 Dec, 2018 17:20:07

✓ Run: success

Step Log

	MODULE	DATE/TIME (UTC -5)	LOG LEVEL	MESSAGE
1	ipaas_logger	10 Dec, 2018 17:20:07, 530	FLOW	Start Query IS Manager
2	ipaas_logger	10 Dec, 2018 17:20:07, 531	FLOW	Collecting diagnostics from primary manager node None
3	ipaas_logger	10 Dec, 2018 17:20:07, 533	FLOW	Trying to connect to None (1/3)
4	ipaas_logger	10 Dec, 2018 17:20:17, 550	FLOW	Trying to connect to None (2/3)
5	QueryISSwarmManager	10 Dec, 2018 17:20:27,	ERROR	Unable to query Swarm manager due to error: Could not connect to None. Giving up. Swarm related diagnostics may not be available in the report

TIP: Click the gray area of the **Logs** pane to close the pane.

Chapter

7

API Endpoints in the Integration Service

Overview

The Integration Service includes an API that is available after you install the Integration Service system.

This chapter covers the following topics:

Interacting with the API

To view the full documentation for the IS API:

1. From the Integration Service system, copy the file **/opt/iservices/scripts/swagger.yml** to your local computer.
2. Open a browser session and go to editor.swagger.io
3. In the Swagger Editor, open the **File** menu, select **Import File**, and import the file swagger.yml. The right pane in the Swagger Editor displays the IS API documentation.

Available Endpoints

POST

/applications. Add a new application or overwrite an existing application.

/applications/run. Run a single application by name.

/configurations. Add a new configuration or overwrite an existing configuration.

/steps. Add a new step or overwrite an existing step.

/steps/run. Run a single step by name.

/schedule. Add a new schedule entry.

GET

/applications. Retrieve a list of all available applications.

/applications/{appName}. Retrieve a specific application.

/applications/{appName}/logs. Retrieve the logs for the specified application.

/cache/{cache_ID}. Retrieve a specific cache.

/configurations. Retrieve a list of all configurations available.

/configurations/{configName}. Retrieve a specific configuration.

/reports. Retrieve a list of all available reports.

/reports/{appName}. Retrieve a specific report by name.

/reports/{reportId}. Retrieve a specific report by ID.

/steps. Retrieve a list of all steps.

/steps/{stepName}. Retrieve a specific step.

/schedule. Retrieve a list of all schedule entries.

/tasks/{taskId}. Retrieve a specific task.

REST

/tasks. Terminate all running tasks.

/tasks/{taskId}. Terminate a specific running task.

DELETE

/schedule/{scheduleName}. Delete a schedule entry by ID.

/reports/{appName}. Delete a specific report by name.

/reports/{reportId}. Delete a specific report by its ID.

Chapter

8

Using SL1 to Monitor the Integration Service

Overview

This manual describes how to monitor the Integration Service in SL1 using the *ScienceLogic: Integration Service* PowerPack.

The following topics provide an overview of the *ScienceLogic: Integration Service* PowerPack:

NOTE: ScienceLogic provides this documentation for the convenience of ScienceLogic customers. Some of the configuration information contained herein pertains to third-party vendor software that is subject to change without notice to ScienceLogic. ScienceLogic makes every attempt to maintain accurate technical information and cannot be held responsible for defects or changes in third-party vendor software. There is no written or implied guarantee that information contained herein will work for all third-party variants. See the End User License Agreement (EULA) for more information.

What Does the Integration Service PowerPack Monitor?

To monitor an Integration Service instance with SL1, you must install the *ScienceLogic: Integration Service* PowerPack. This PowerPack lets you configure SL1 to create an alert if an integration application in the Integration Service fails.

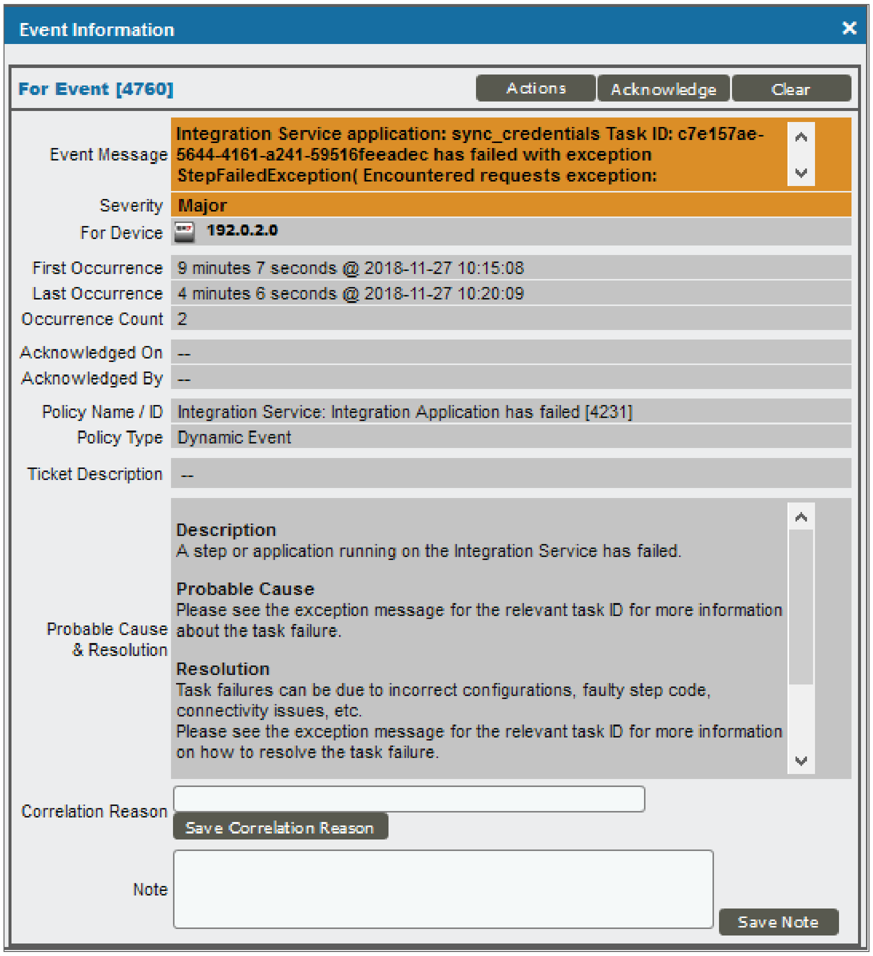
The *ScienceLogic: Integration Service* PowerPack includes:

- The "ScienceLogic: Integration Service Queue Configuration" Dynamic Application, which monitors the status of the Integration Service RabbitMQ service
- This PowerPack also includes the "REST: Performance Metrics Monitor" Dynamic Application, which monitors outgoing REST requests from SL1 to the Integration Service
- Event Policies and corresponding alerts that are triggered when an integration application in the Integration Service fails
- A Device Class for the Integration Service
- A Sample Credential for connecting to the Integration Service
- ScienceLogic Libraries that are utilized by this PowerPack:
    - content
    - content_cache
    - silo_core
    - silo_core_rest
    - silo_credentials

Monitoring Integration Applications in the Integration Service

The *ScienceLogic: Integration Service* PowerPack includes the ability to create an alert in the associated SL1 system if an integration application in the Integration Service fails.

The "ScienceLogic: Integration Service Queue Configuration" Dynamic Application generates a Major event in SL1 if an integration application fails in the Integration Service:



Event Message	Integration Service application: sync_credentials Task ID: c7e157ae-5644-4161-a241-59516feeadec has failed with exception StepFailedException(Encountered requests exception:
Severity	Major
For Device	192.0.2.0
First Occurrence	9 minutes 7 seconds @ 2018-11-27 10:15:08
Last Occurrence	4 minutes 6 seconds @ 2018-11-27 10:20:09
Occurrence Count	2
Acknowledged On	--
Acknowledged By	--
Policy Name / ID	Integration Service: Integration Application has failed [4231]
Policy Type	Dynamic Event
Ticket Description	--
Probable Cause & Resolution	**Description** A step or application running on the Integration Service has failed. **Probable Cause** Please see the exception message for the relevant task ID for more information about the task failure. **Resolution** Task failures can be due to incorrect configurations, faulty step code, connectivity issues, etc. Please see the exception message for the relevant task ID for more information on how to resolve the task failure.
Correlation Reason	
Note	

The related Event Policy includes the name of the application, the Task ID, and the traceback of the failure. You can use the application name to identify the integration application that failed on the Integration Service. You can use the Task ID to determine the exact execution of the application that failed, which you can then use for debugging purposes.

To view more information about the execution of an application in the Integration Service, navigate to the relevant page in the Integration Service by formatting the URL in the following manner:

```
https://<integration_service_hostname>/integrations/<application_name>?runid=<task_
id>
```

For example:

```
https://192.0.2.0/integrations/sync_credentials?runid=c7e157ae-5644-4161-a241-
59516feeadec
```

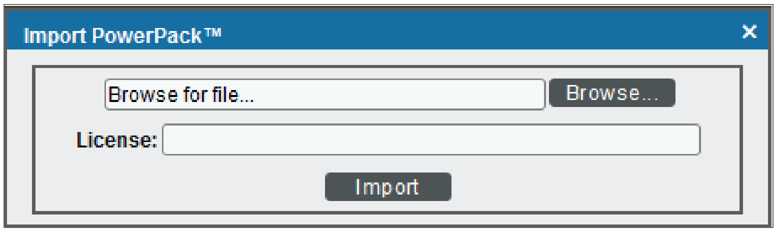
Installing the ScienceLogic: Integration Service PowerPack

Before completing the steps in this section, you must import and install the latest version of the *ScienceLogic: Integration Service* PowerPack.

TIP: By default, installing a new version of a PowerPack overwrites all content in that PowerPack that has already been installed on the target system. You can use the ***Enable Selective PowerPack Field Protection*** setting in the **Behavior Settings** page (System > Settings > Behavior) to prevent new PowerPacks from overwriting local changes for some commonly customized fields. (For more information, see the ***System Administration*** manual.)

To download and install a PowerPack:

1. Download the PowerPack from the ScienceLogic Customer Portal.
2. Go to the **PowerPack Manager** page (System > Manage > PowerPacks).
3. In the **PowerPack Manager** page, click the **[Actions]** button, then select *Import PowerPack*.
4. The **Import PowerPack** dialog box appears:



5. Click the **[Browse]** button and navigate to the PowerPack file.
6. When the **PowerPack Installer** modal page appears, click the **[Install]** button to install the PowerPack.

NOTE: If you exit the **PowerPack Installer** modal page without installing the imported PowerPack, the imported PowerPack will not appear in the **PowerPack Manager** page. However, the imported PowerPack will appear in the **Imported PowerPacks** modal page. This page appears when you click the **[Actions]** menu and select *Install PowerPack*.

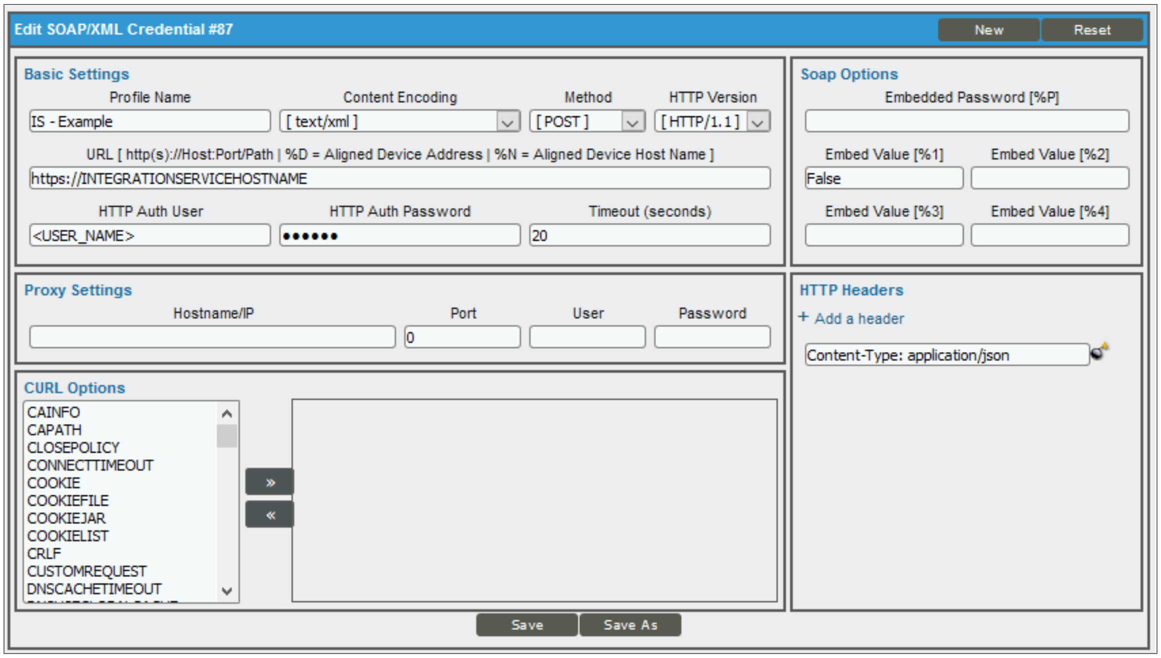
Creating a SOAP/XML Credential for the Integration Service

To configure SL1 to monitor the Integration Service, you must first create a SOAP/XML credential. This credential allows the Dynamic Applications in the *ScienceLogic: Integration Service* PowerPack to communicate with the Integration Service.

The PowerPack includes an example SOAP/XML credential that you can edit for your own use.

To configure a SOAP/XML credential to access the Integration Service:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Locate the **IS - Example** credential, and then click its wrench icon (🔧). The **Edit SOAP/XML Credential** modal page appears:



Edit SOAP/XML Credential #87

- Profile Name: IS - Example
- Content Encoding: [text/xml]
- Method: [POST]
- HTTP Version: [HTTP/1.1]
- URL [http(s)://Host:Port/Path | %D = Aligned Device Address | %N = Aligned Device Host Name]: https://INTEGRATIONSERVICEHOSTNAME
- HTTP Auth User: <USER_NAME>
- HTTP Auth Password: ••••••
- Timeout (seconds): 20
- Embed Value [%1]: False
- Proxy Port: 0
- HTTP Headers: Content-Type: application/json

3. Complete the following fields:

   - ***Profile Name***. Type a name for the Integration Service credential.
   - ***Content Encoding***. Select *text/xml*.
   - ***Method***. Select *POST*.
   - ***HTTP Version***. Select *HTTP/1.1*.
   - ***URL***. Type the URL for your Integration Service system.
   - ***HTTP Auth User***. Type the Integration Service administrator username.
   - ***HTTP Auth Password***. Type the Integration Service administrator password

- ***Timeout (seconds)***. Type "20".
- ***Embed Value [%1]***. Type "False".
- ***HTTP Headers***. Use the attached "Content-Type: application/json" header or click **Add a header** to define a custom HTTP header. Click the "bomb" icon to remove a header you do not need.

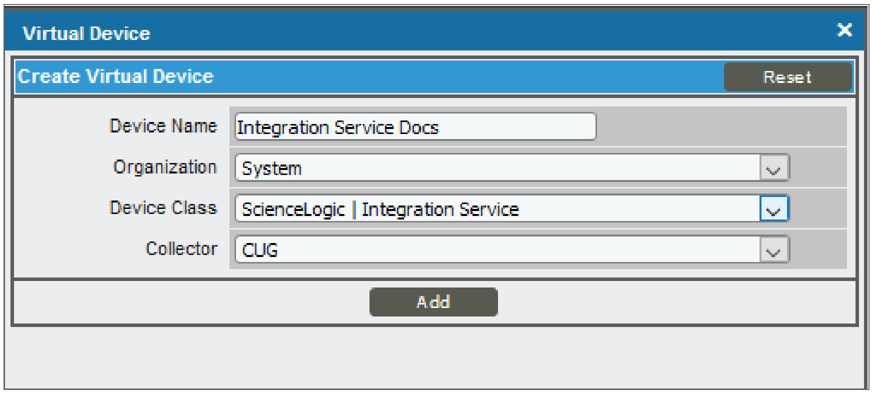
4. Click the **[Save As]** button.

Creating a Virtual Device for the Integration Service PowerPack

To monitor the Integration Service, you must create a ***virtual device*** that represents the Integration Service. You can use the virtual device to store information gathered by policies or Dynamic Applications.

To create a virtual device that represents the Integration Service:

1. Go to the **Device Manager** page (Registry > Devices > Device Manager).
2. Click the **[Actions]** button and select *Create Virtual Device* from the menu. The **Virtual Device** modal page appears:



Device Name: Integration Service Docs
Organization: System
Device Class: ScienceLogic | Integration Service
Collector: CUG

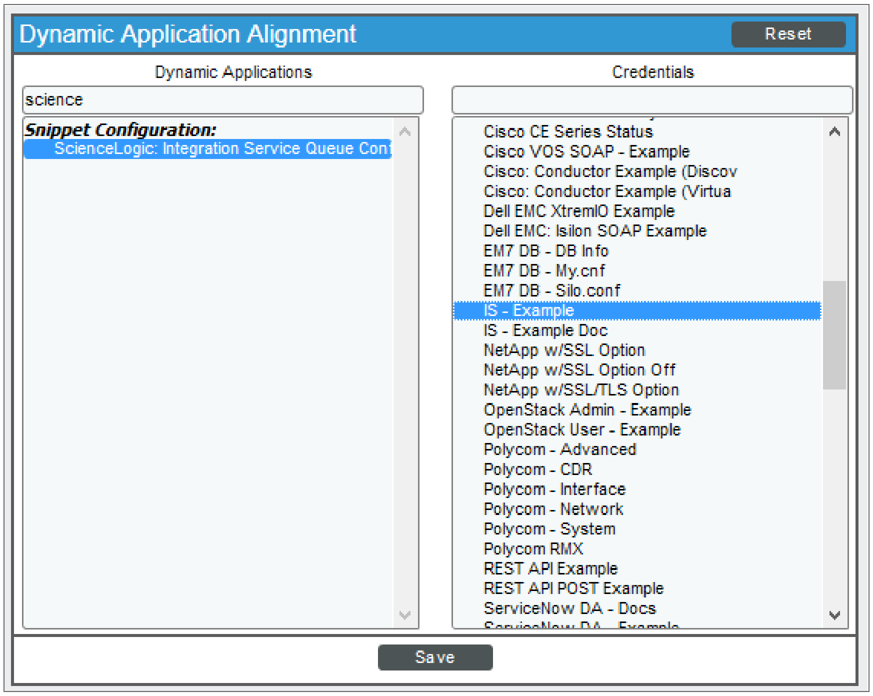
3. Complete the following fields:
   - ***Device Name***. Type a name for the device.
   - ***Organization***. Select the organization for this device. The organization you associate with the device limits the users that will be able to view and edit the device. Typically, only members of the organization will be able to view and edit the device.
   - ***Device Class***. Select *ScienceLogic | Integration Service*.
   - ***Collector***. Select the collector group that will monitor the device.
4. Click the **[Add]** button to create the virtual device.

Aligning the Integration Service PowerPack Dynamic Applications

Before you can run the Dynamic Applications in the *ScienceLogic: Integration Service* PowerPack, you must manually align each Dynamic Application to the virtual device you created in the previous step. Use the credential based on the **IS - Example** credential when you align the Dynamic Applications.

To align the Integration Service Dynamic Applications with the Integration Service virtual device:

1. Go to the **Device Manager** page (Registry > Devices > Device Manager).
2. Click the wrench icon (🔧) for the virtual device you created in the previous step. The **Device Properties** page appears.
3. Click the **[Collections]** tab. The **Dynamic Application Collections** page appears.
4. Click the **[Actions]** button and select *Add Dynamic Application*. The **Dynamic Application Alignment** modal page appears:



5. In the ***Dynamic Applications*** field, select the first of the Integration Service Dynamic Applications.
6. In the ***Credentials*** field, select the credential you created based on **IS - Example** credential.
7. Click the **[Save]** button.
8. Repeat steps 4-7 for each remaining Dynamic Application.

LIMITATION OF LIABILITY AND GENERAL DISCLAIMER

ALL INFORMATION AVAILABLE IN THIS GUIDE IS PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED. SCIENCELOGIC™ AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT.

Although ScienceLogic™ has attempted to provide accurate information on this Site, information on this Site may contain inadvertent technical inaccuracies or typographical errors, and ScienceLogic™ assumes no responsibility for the accuracy of the information. Information may be changed or updated without notice. ScienceLogic™ may also make improvements and / or changes in the products or services described in this Site at any time without notice.

Copyrights and Trademarks

ScienceLogic, the ScienceLogic logo, and EM7 are trademarks of ScienceLogic, Inc. in the United States, other countries, or both.

Below is a list of trademarks and service marks that should be credited to ScienceLogic, Inc. The ® and ™ symbols reflect the trademark registration status in the U.S. Patent and Trademark Office and may not be appropriate for materials to be distributed outside the United States.

- ScienceLogic™
- EM7™ and em7™
- Simplify IT™
- Dynamic Application™
- Relational Infrastructure Management™

The absence of a product or service name, slogan or logo from this list does not constitute a waiver of ScienceLogic's trademark or other intellectual property rights concerning that name, slogan, or logo.

Please note that laws concerning use of trademarks or product names vary by country. Always consult a local attorney for additional guidance.

Other

If any provision of this agreement shall be unlawful, void, or for any reason unenforceable, then that provision shall be deemed severable from this agreement and shall not affect the validity and enforceability of any remaining provisions. This is the entire agreement between the parties relating to the matters contained herein.

In the U.S. and other jurisdictions, trademark owners have a duty to police the use of their marks. Therefore, if you become aware of any improper use of ScienceLogic Trademarks, including infringement or counterfeiting by third parties, report them to Science Logic's legal department immediately. Report as much detail as possible about the misuse, including the name of the party, contact information, and copies or photographs of the potential misuse to: legal@sciencelogic.com

800-SCI-LOGIC (1-800-724-5644)

International: +1-703-354-1010